Fall 1-31-1997

Modeling of objects using planar facets in noisy range images

Girish Phansalkar
New Jersey Institute of Technology

Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

ABSTRACT

MODELING OF OBJECTS USING PLANAR FACETS IN NOISY RANGE IMAGES

by
Girish Phansalkar

Products designed and manufactured before the advent of Computer Aided Design (CAD) and Computer Aided Manufacturing (CAM) technology have not been documented electronically. To avoid the laborious procedure of redesigning the parts, a *reverse engineering* approach can be adopted. This approach involves, taking a picture of the object and constructing a solid model from the image data.

Range image is a three dimensional image of an object or a scene. This image can be obtained from special cameras, called range image cameras, or can be constructed from the Coordinate Measuring Machine's (CMM) output data. *Adaptive Fuzzy c-Elliptotype* (AFC) clustering algorithm is used to identify the planar facets in a range image. A modified version of AFC algorithm can handle noisy range images. Unknown number of planar facets can be identified using the *Agglomerative* clustering approach.

The object is reconstructed using segmented image data. The equations of the edge are obtained from the plane intersections. An *edge validity criterion* is developed to validate the existence of an edge. Vertices are the two extreme points on the edge. A Boundary representation of the object is developed. The information about this object is then passed to a CAD software using *Initial Graphics Exchange Specification* (IGES).

MODELING OF OBJECTS USING PLANAR FACETS IN NOISY RANGE IMAGES

by
Girish Phansalkar

A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Mechanical Engineering

Department of Mechanical Engineering

APPROVAL PAGE

MODELING OF OBJECTS USING PLANAR FACETS IN NOISY RANGE IMAGES

Girish Phansalkar

Dr. Rajesh N. Dave, Thesis Advisor — Date
Associate Professor of Mechanical Engineering, NJIT

Dr. Reggie Caudill, Committee Member — Date
Professor of Mechanical Engineering, NJIT

Dr. Zhiming Ji, Committee Member — Date
Associate Professor of Mechanical Engineering, NJIT

BIOGRAPHICAL SKETCH

Author: Girish Phansalkar

Degree: Master of Science

Date: January 1997

Undergraduate and Graduate Education:

- Master of Science in Mechanical Engineering
  New Jersey Institute of Technology, Newark, NJ, 1997

- Bachelor of Technology in Mechanical Engineering
  Indian Institute of Technology, Bombay, India, 1995

Major: Mechanical Engineering

Publications:

Phansalkar G. and Dave R.N., "On Generating Solid Models of Mechanical Parts through Fuzzy Clustering", submitted, *Sixth IEEE Conference on Fuzzy systems, (FUZZ-IEEE'97),* Barcelona, Spain, July 1-5, 1997.

This thesis is dedicated to
my beloved parents

ACKNOWLEDGMENT

I would like to express my sincere gratitude to Dr. Rajesh Dave, who not only helped me as my thesis adviser, providing valuable resources, insights and intuition, but also gave me support, encouragement and reassurance.

Special thanks to Dr. Reggie Caudill and Dr. Jhiming Ji for actively participating in my committee.

I also wish to express my sincere thanks to Dr. Anthony Rosato for encouraging me in academic and extra-curricular activities and providing the facilities of the Particle Technology Center for my thesis.

Finally, I thank Vivek Gupta for his help and support over the years.

CHAPTER 1
INTRODUCTION

1.1 Introduction to Solid Modeling

A solid geometric model is the unambiguous and informationally complete mathematical representation of a shape of a physical object in the form that computer can process. Solid modeling is an important aspect of geometric modeling that is used to create and communicate shape information. It involves creating and maintaining a representation of an object for future access, update and analysis.

The parts designed and manufactured before the advent of the *Computer Aided Design & Computer Aided Manufacturing* (CAD/CAM) technology have not been designed using computers. Electronic solid models of such parts are generally not available. Nowadays, the use of computerized product data management systems is gaining increasing importance. This system allows effective product data handling and instant access to this data from remote places. Hence, it is important to generate solid models of the old non-documented parts. This could be done by re-drafting the parts using the CAD software. But, this method is often time consuming and laborious. If an organization has large number of such parts, then lot of time and money would be spent on constructing the solid models. To get over this problem a *reverse engineering* approach has been proposed.

Reverse engineering involves reconstructing the solid models from the image of the object. However, it is much more involved than using the latest digitizing technology to capture complex free-form surfaces. A 3-D laser digitizing system scans the surface of the objects to generate a *range image*. A range image is a set of data points on the surface of an object, represented in a 3-D Cartesian co-ordinate system. Range image cameras are currently being used in space and defense applications. The economic viability of these cameras in manufacturing environment needs to be studied. However, they are expected to be routinely used in manufacturing applications in the near future. Range images can also be constructed from different 2D views of the

same object or by a Co-ordinate Measuring Machine (CMM). Digitizing, however, is only a part of the process and it is the part that requires least amount of time and effort. The bulk of the effort lies in manipulating the digitized data into a usable form.

The range image must be processed to generate the desired mathematical model. This task can be divided into two broad categories:

1. The range image data obtained from a laser digitizer is a cloud of data points. Hence to identify different facets of the object, it is divided into a number of clusters. This process is called *data segmentation* or partitioning of data. However, if the range image is obtained from a CMM, the operator could segment the data manually. ie. the operator of CMM can store the data points belonging to different facets in different files. This would make it unnecessary to partition the data further, as the data set would be already partitioned. However, to automate the reverse engineering system, it is desired to partition the data automatically. Hence, the first step in this process is segmentation of the range image.
2. To construct a mathematical model of an object the partitioned range image is analyzed to generate surface patches, ie. the surfaces are modeled mathematically. Geometric solid model of the object is then constructed from the surface information.

The thesis aims at proposing, evaluating and improving algorithms for segmentation of range image into planar clusters. Planar segmentation and plane fitting is easy to handle and hence has been adopted to accomplish the task. A *boundary representation* (B-Rep) of the object is proposed.

As mentioned earlier, the usual format of a range image is such that for each point in the data set, there is an associated position vector in a three dimensional Cartesian space. Therefore, the problem of classifying these points into meaningful clusters, is infact, a clustering problem of unlabelled data points.

Note that in the following discussion, the words "object" and "part" have been used synonymously. An object means a physical object unless explicitly specified.

1.2 Introduction to Cluster Analysis

Clustering of a data set X means dividing X into c clusters, such that $2 \leq c < n$, where n is the number of data points in the set. The points are clustered depending on some common mutual relationship. The structure of this cluster is dictated by the type of relationship. The point is classified into a cluster i if its relationship with this cluster is stronger than its relationship with any other cluster j where $j \neq i$. A measure has been established to quantify this relationship. This measure, also called the clustering criterion, is typically some mathematical property of the data set like distance from a cluster prototype. The clustering criterion decides the shape, size and other physical properties of the clusters identified. A well defined clustering criterion supplemented by an effective clustering method give good clustering results.

The main classes of clustering algorithms are :

- Hierarchical
- Graph-Theoretic
- Objective Function

The memberships of the points in the clusters could be either hard or fuzzy. In hard memberships the data point can only be assigned to one cluster. ie. the memberships are either 0 or 1. In fuzzy approach, the membership of a point in a cluster varies from 0 to 1. The relative memberships decide the cluster in which the particular point falls.

The fuzzy clustering approach can thus be used to split the range image into various clusters, each of which represents the surface of the object.

1.3 Literature Survey

If a machine part fails, the production line stops due to the crucial role played by the machine. If no spare parts for this machine are available, the broken part is given to the machinist, a new part is made and the machine resumes the normal service. In this case

no drawings of the part are available. A procedure to recreate such parts is implemented in many organizations is called reverse engineering.

Reverse Engineering is systematic evaluation of a product with a purpose of replication. It may involve making exact copy of a part or making changes in it to suit the need. Reverse engineering is useful in following circumstances [4]:

1. *Old Products*: Products designed before CAD/CAM existed have to be documented.
2. *Products Modified*: Products modified during use need to be designed.
3. *Design Reuse*: To design a new product, a manufacturer uses the old designs and modifies them. In this case reverse engineering comes in handy.
4. *Overseas Products*: The overseas products which have no documentation can be documented using reverse engineering.

Reverse Engineering has undergone mutations changing from a skilled manual process to an engineering tool using sophisticated software.

Graphite drawings and blueprints have been replaced by CAD systems for storage and modification of product definition data. New applications of the computer models like rapid prototyping are being developed to improve the design, testing and manufacturing capabilities. CAD models allow greater flexibility in modification and reproduction. CAD/CAM systems also improve communication between different groups involved in the product development. The research in reverse engineering is thus propelled towards generating CAD models from the mechanical parts [2].

The parts that have been used over a period of time may have warped or changed in shape over a period of time. Hence, an exact replica of a part may not actually represent its original design. However, it is the closest approximation. So, the reverse engineering technique aims at producing the exact replica.

The standard practice in reverse engineering has been to digitize the points on the surface of the part and transfer them to the CAD system.

The part digitizing techniques can be classified into two main categories[24]:

- Contact Technique

- Non-contact Technique

Contact technique is the most commonly used technique. It makes use of Co-ordinate Measuring Machine (CMM) that is mainly used for part inspection in factories. Nowadays some of the co-ordinate measuring machines come with a specialized software for digitizing parts. There are some other contact type techniques used for the measurement. A lighted pen, a low frequency electromagnetic transducers and an ultrasonic imaging are only some of these techniques. Here, the emitter touches the part and the position of the emitter is detected. The non-contact techniques are:

1. *Structured Lighting Technique*: There is one light source to illuminate the part. Detector is situated at a different view port. The light source can be single beam of light, multiple stripes of light, or a coded grid of light like lasers [17].
2. *Spot Ranging*: This uses the time of flight or the frequency modulation of the collimated source of light or ultra sound to detect the position of the point on a part surface.
3. *Range from Focus*: In a camera, the focusing distance can be used to determine the depth of a point on a part surface.
4. *Range from Texture*: In capturing the image, the perceived fineness of texture increases with increased distance from the range image. If the object texture has been calibrated this measure could be used to get depth information.
5. *Stereo Scanning*: Two or more views of same object can be analyzed and depth information can be obtained by triangulation.

Contact type techniques are more accurate. But, they are extremely slow. In contrast, non-contact type techniques are extremely fast and can digitize thousands of points in a second. Non-contact scanning is desirable in case of soft or flexible bodies so that the body is not deformed while scanning. Out of all the techniques mentioned above the non-contact type CMM technique is most popular in industries but the laser technique is also gaining ground.

Digitized data that is obtained is called a point-data cloud. Surface features have to be extracted out of this cloud. Operator intelligence is still required when

multiple surfaces are involved [4]. The thesis typically addresses this problem in reverse engineering. Some attempts at generating a tessellated model have been made. However, these methods do not represent the surfaces in their original form but approximate them by small patches like triangles. Motavalli and Bidanda [24] have tackled the problem for parts with rotational symmetry. "Surfacer", a software developed by Imageware[1], generates surfaces from data points. But the process is still not fully automated.

The fuzzy clustering has a capability of identifying clusters of points with common attributes. This could be effectively used for surface approximation [21]. The surfaces could then be transferred to a CAD software. The surfaces could also be used to construct solid models.

The solid model is a most complete and closest possible representation a physical object. For any given point in space, a CAD/CAM software can tell whether the point is inside or outside the object if the object is represented as a solid model. Also, most of the physical properties such as mass, moment of inertia, volume etc. can be calculated using solid models.

There are various ways of solid modeling. Some of them are rarely used but are mentioned for the sake of completeness [23].

1. *Cell Decomposition*: The solid is represented as a sum or union of a set of cells into which it is divided. The total object may not be amenable to representation but its cells are. This is regularly used in structural analysis problems. This representation is not unique but is unambiguous.
2. *Spatial Occupancy Enumeration*: The cells are cubical and are located in a grid in a 3-D space. The cells occupied by the object are enumerated. As the size of the cells decreases the representation approaches a set of contiguous points in space. However, this method requires enormous amount of data to be handled.
3. *Octree Representation*: Octree representation is an efficient way of spatial enumeration in which the parts of the body occupying adjacent positions are fused together if they form an identical representation but larger in size[3].

4. *Sweep Representations*: In this representation, a generator surface is swept along a director curve. There are two principal types of trajectories: translational and rotational.[1]
5. *Constructive Solid Geometry* (CSG): This modeling method involves defining complex solids as a combination of simpler solids called primitives. The complex solid is obtained by regularized Boolean operations on the primitives and their co-ordinate transformations. This representation is extensively used because of ease of use and minimal storage requirements.
6. *Boundary Representation* (B-Rep): In this case, an object is defined by the volume enclosed by a set of closed bounding surfaces [15,23]. Since this representation has explicit surface information it is useful for extracting information about the properties where surface interaction is involved.

The geometric model generated from a point data cloud is then transferred to a commercial CAD package. CAD softwares are equipped with many applications like testing, drafting, machining, and rapid prototyping. A part model recreated using a reverse engineering approach can these applications. Hence, the model needs to be transferred to a CAD software. Developing a specialized translator for each of the software is tedious and impractical. The data communication between the CAD software usually occurs through neutral files. These files are written in accordance to some standards. *Initial Graphics Exchange Specification* (IGES) is an ANSI standard which most of the softwares can recognize. It was developed by US Product Data Association (USPRO). IGES representation of geometric and physical entities is independent of the software. Each software is equipped with a translator to read and write in IGES format. IGES can, thus, be termed as a language for digital data exchange between the softwares [1]. The use of IGES is thus recommended for transferring a geometric model to a CAD software [7, 18, 22].

---

[1] Imageware Inc., Ann Arbor, MI.

Some problems, however, have been noticed while transferring data about objects with complex surfaces. This problem is especially encountered in case of free form surfaces. eg. while transferring a solid model of an apple to Pro/ENGINEER® only surface model is transferred. However, it is still the best digital data transfer specification available. Being an ANSI standard, it will be used for data transfer. The problem mentioned above is not encountered here, as only objects with planar facets have been modeled.

1.4 Objective

The objective of this thesis is to develop a 3-D object recognition system. This involves segmentation of the range image using fuzzy clustering approach. Algorithms for handling noise in range images is known to work. The problem of handling unknown number of clusters needs to be developed.

The solid model of the object is constructed using the planar features. A boundary representation of the object is the easiest to construct from the surface data. Plane fitting will be done on the segmented data. The faces are completely defined by the equations of the planes and their bounding edges. The equations of the bounding edges will be defined by the intersection of planes. An *edge validity criterion* needs to be formulated to validate the existence of the edges. The data generated from this system should be exported to an IGES file. This IGES file can be read by any commercial CAD software to construct the solid model.

The following chapters will discuss the proposed reverse engineering technique in detail. Chapter 2 will discuss various methods of range image segmentation and their relative performance. In chapter 3, methods of plane fitting will be discussed. Edge detection, validation, trimming and linking will be the topic of discussion in chapter 4. Chapter 5 will cover the solid modeling aspects in detail along with discussion of results. Finally, chapter 6 will conclude the topic.

CHAPTER 2

FUZZY CLUSTERING USING OBJECTIVE FUNCTION

2.1 Fuzzy c-Means (FCM) Algorithm

Following the notations and definitions of Bezdek [6] the basic Fuzzy c-Means algorithm can be described as follows:

The FCM minimizes an objective functional J_m: $M_{fc} \times R^{cp} \rightarrow R^+$

$$J_m(U,v) = \sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ik})^m(d_{ik})^2 \qquad (2.1.1)$$

where

M_{fc} is the fuzzy partition space

R^{cp} are the c p-tuples of the real numbers.

$U \in M_{fc}$ is a fuzzy c-partition of data set **X**

$(v=(v_1, v_2, .. v_c))$ is the cluster center vectors

d_{ik} is given by

$$(d_{ik})^2 = \|x_k - v_i\|^2 \qquad (2.1.2)$$

and $\|\bullet\|$ is any inner product induced norm on R^p;

$m \in [1,\infty)$ is the weighting exponent or the fuzzifier.

In the above equations, index i denotes one each of c number of clusters, p the dimension of R space from which the data is derived, and index k one each of n number of data-points. It can be realized that J_m is a squared error clustering functional, and solutions that minimize J_m are least-squared error stationary points of J_m. The basic procedure to find the solution is Picard iteration, consisting of following steps.

Algorithm 1(FCM)

1. Fix c, $2 \le c < n$; choose any inner product norm; fix m, $1 \le m < \infty$; initialize the membership matrix $U^{(0)}$.
2. Calculate c fuzzy cluster centers v_i as

$$v_i = \sum_{k=1}^{n} (u_{ik})^m x_k \Big/ \sum_{k=1}^{n} (u_{ik})^m \tag{2.1.3}$$

3. Update membership function $U^{(i)}$ to $U^{(i+1)}$ as

$$u_{ik} = 1 \Big/ \left[\sum_{j=1}^{c} \left(d_{ik} / d_{jk} \right)^{2/(m-1)} \right] \tag{2.1.4}$$

4. Compare the change in the membership values using a convenient norm; stop if

$$\left| U^{(i+1)} - U^{(i)} \right| < \varepsilon \tag{2.1.5}$$

Else $i = i+1$ and return to 2.

This algorithm tends to find clusters that are spherical in shape. Although it may not be directly used for line or plane detection in an image space, it may be used in a feature space to detect linear shapes. For example, in a range image segmentation, the orientation and curvature may be used as features, and then FCM may be used to partition the data [10].

2.2 Fuzzy c-Varieties (FCV) Clustering Algorithms

This variation allows the c prototypes to be r-dimensional linear varieties, $0 \le r \le p-1$, rather than just points (r=0) as in FCM algorithms. Thus, distances in such algorithms are measured from lines (r=1), planes (r=2), or hyper-planes (r>2). This infinite family of algorithms[6, 10] can detect clusters of r-dimensional linear shapes. The

FCV with r=1 & r=2 are called Fuzzy c-Lines (FCL) and Fuzzy c-Planes (FCP) respectively. These algorithms may be used to detect lines and planes. However, if the prototype is an infinite line or an infinite plane, it may tend to pick up very large clusters. Figure 2.1 shows an incomplete range image of a V shaped block henceforth referred to as "jig". Figure 2.1 shows this range image after segmenting it using FCV algorithm. As shown, FCV tries to find infinite planes and thus the results are unsatisfactory. FCV algorithm may be improved by a following variation.

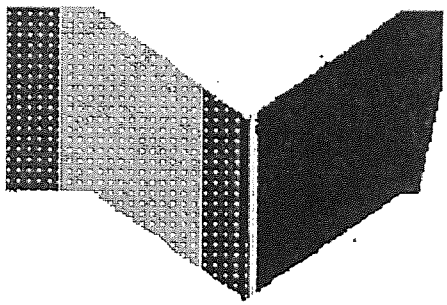


Figure 2.1 Range image of a jig segmented by FCV (r=2)
n = 19400, c = 3, ε = 0.001

2.3 Fuzzy c-Elliptotypes (FCE) Clustering Algorithms

These algorithms are a further generalization of FCM and FCV through their convex combinations. These algorithms seek clusters of elliptical shapes in the data. Convex combination is achieved, for example, in the case of combination of FCM and FCL, by defining a new functional J_e as

$$J_e(U,v)=(1-\alpha)J_m+\alpha J_l \qquad (2.3.1)$$

where α is the mixing coefficient, J_m is the FCM functional as defined in equation (2.1.1), and J_l is the functional for FCL, defined below.

$$J_l(U,v)=\Sigma\Sigma(u_{ik})^m(D_{ik})^2 \tag{2.3.2}$$

D_{ik} in the above equation is the distance of the point from the linear prototype (line), and can be given by

$$(D_{ik})^2=(d_{ik})^2-(<x_k-v_i,d_i>)^2 \tag{2.3.3}$$

where d_i is the direction of the i^{th} cluster (line).

The functional can also be written in the following form:

$$J_{el}(U,v)=\Sigma\Sigma(u_{ik})^m(Z_{ik})^2 \tag{2.3.4}$$

where Z_{ik} is a modified distance given by

$$(Z_{ik})^2 = \alpha(D_{ik})^2 + (1-\alpha)(d_{ik})^2 \tag{2.3.5}$$

The mixing coefficient α, then defines the proportion by which the distance is measured from a point and a line, thus defining ellipsoidal clusters. Similar convex combination can be achieved between FCM and FCP through the following equation.

$$J_e(U,v)=(1-\alpha)J_m+\alpha J_p \tag{2.3.6}$$

where J_p is the functional for FCP.

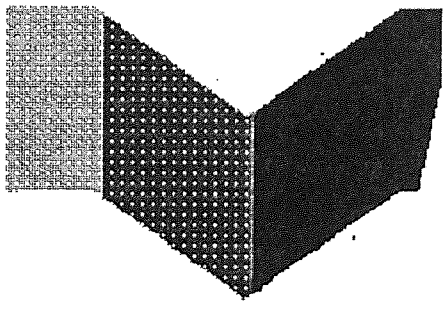


Figure 2.2 Range image of a jig segmented using FCE algorithm
n =19400, c = 3, ε = 0.001

The mixing coefficient α restricts the size of the line or the plane to be identified. If this number is slightly less than unity then we can find out finite linear or planar clusters. However, this is not a comprehensive solution to the problem [12]. Since the same value of α is used for all the clusters, the algorithm will seek clusters of the same elliptical or ellipsoidal shape, and therefore may not work when the clusters are of different sizes. This problem was solved by a heuristic technique to adaptively select the mixing coefficient α [9, 12]. The algorithm based on this approach is called the Adaptive Fuzzy c-Varieties (AFC) algorithm.

2.4 Adaptive Fuzzy c-Elliptotypes (AFC) Clustering Algorithms

The objective functional J_a for the AFC algorithm is defined in the same fashion as J_e in equation (2.3.3). Dave [5, 12] and Gunderson [16] have addressed the problem of deciding α. In the approach used by Dave, the modified distance Z_{ik} is defined differently for each cluster. Here, only the combination of FCM and FCL is considered. However, the combination of FCM and FCP is similar.

$$(Z_{ik})^2=\alpha_i (D_{ik})^2 + (1-\alpha_i)(d_{ik})^2 \qquad (2.4.1)$$

The mixing coefficients α_i are defined as

$$\alpha_i = 1 - (\beta_i / \gamma_i) \qquad (2.4.2)$$

with $\beta_i = \min (\lambda_{1i}, \lambda_{2i})$ and $\gamma_i = \max (\lambda_{1i}, \lambda_{2i})$, $\qquad i = 1, .. , c.$

In the above equations, λ_{ji} are the eigenvalues of the fuzzy scatter matrix of cluster i. The AFC algorithms can be used to detect lines or planes in a data-set. Examples of line detection are considered in Dave[15]. In this approach, the equation (2.4.2) is derived from the following argument. A spherical cluster should have α=0 and a perfectly planar infinite cluster should have α=1. The orthogonal eigenvectors of

the scatter matrix should be directed along the major and the minor axis of the elliptical entity. For an infinite line one eigenvalue should be close to zero and other should be very large. Whereas, for a spherical cluster the two eigenvalues should be equal. Hence, larger ratio of the two eigenvalues indicates a spherical cluster.

All the algorithms considered above are useful in linear boundary detection in one way or another. They may also be used indirectly to detect curved boundaries, provided that the features of curved boundaries can be modeled as a linear prototype. However, they are not directly applicable for curved boundary detection.

A similar approach for detecting surface entities calls for a combination of FCM and FCP algorithms and the new functional will be called J_{a_mp}.

Here,

$$(Z_{ik})^2=\alpha_i (\boldsymbol{D}_{ik})^2 + (1-\alpha_i)(d_{ik})^2 \qquad (2.4.3)$$

In the above equation, distance $\boldsymbol{D}_{ik}$ is the distance of the point from the plane, defined as

$$(\boldsymbol{D}_{ik})^2=(d_{ik})^2-(<x_k-v_i, d_{i1}>)^2-(<x_k-v_i, d_{i2}>)^2 \qquad (2.4.4)$$

where, d_{i1} and d_{i2} are the two principal directions of the eigenvectors of the i^{th} planar cluster corresponding to two largest eigenvalues. The mixing coefficients α_i may be defined as

$$\alpha_i = 1- (\lambda_{1i} /\lambda_{3i}) \qquad (2.4.5)$$

$(\lambda_{1i}, \lambda_{2i}, \lambda_{3i})$ are eigenvalues of the i^{th} fuzzy scatter matrix arranged in ascending order.

The concept of mixing can be further generalized by mixing FCM, FCL and FCP. The new functional will be called J_{a_mlp}. The generalized distance can be specified as

$$(Z_{ik})^2=\alpha_{i1} (\boldsymbol{D}_{ik})^2 +\alpha_{i2} (D_{ik})^2+(1-\alpha_{i1}-\alpha_{i2})(d_{ik})^2 \qquad (2.4.6)$$

assuming distances as in equations (2.3.3) and (2.4.4) and the coefficients defined as

The algorithm in this case shows improved results. It can now identify finite planar clusters of elliptical shape.

Algorithm 2 (AFC)

1. Fix c the number of clusters, $2 \leq c \leq n$, where n is the number of data points;

   Fix fuzzifier $m \in [1,\infty)$

   Initialize the value of mixing coefficients α_{i1}, $\alpha_{i2} \in ((0,1), 1 \leq i \leq c)$

   Initialize membership matrix $U^{(0)}$

For iteration numbers I, I = 1,2...

2. Calculate the c fuzzy centers $\{v_i\}^I$ using memberships $U^{(I)}$
3. Calculate the c fuzzy scatter matrices $\{S_{fi}\}^{(I)}$

$$S_{fi} = \sum_{k=1}^{n} (u_{ik})^m (x_k - v_i)(x_k - v_i)^T \qquad (2.4.8)$$

   Calculate the eigenvalues and the corresponding mixing coefficient.

4. Update $U^{(I)}$ to $U^{(I+1)}$
5. If $(U^{(I+1)}-U^{(I)}) \leq \varepsilon$ stop, else return to 2 with I=I+1

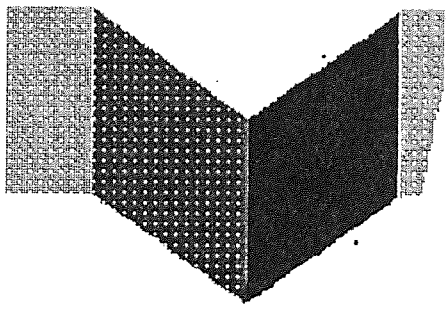


Figure 2.3 Range image of a jig segmented using AFC algorithm
n =19400, c = 3, ε = 0.001

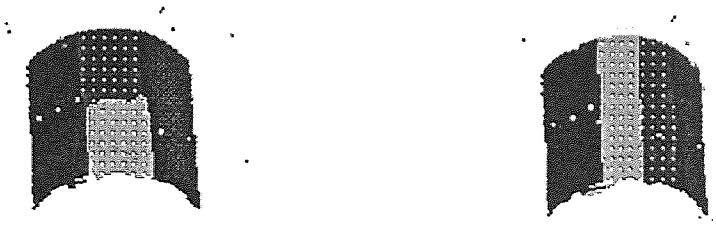


Figure 2.4 Range image of a cylinder segmented by AFC with one α and dual α respectively
$n = 5672$, $c = 4$ $\varepsilon = 0.001$

2.5 Noise in Clustering

The presence of noise is a universal problem in any acquisition device. There are however two types of noise in a data set. Noise arising due to the statistical distribution of the data is deterministic. The distribution of this noise can be forecasted from the accuracy of the measuring instrument. However, there is other type of noise which appears arbitrarily in a data set. The arbitrary distribution of noise is of real concern.

Noise is a common and a serious problem in cluster analysis. The squared error type of algorithms are highly susceptible to noise and their breakdown point is 0. The techniques of cluster analysis described previously do not work well in noisy data. Infact even a very small number of noise points can lead to very bad results. There are three ways of handling the problem of noise in clustering:

1. *Using a Large Value of Fuzzifier m*: It is assumed that when there are a very few noise points their distance from the cluster centers is large. Hence, each of the points will have a small and comparable membership in all the clusters. Hence if this small value is raised to a high power, then its effect in the objective functional will reduce, thus reducing the effect of noise in the clustering. However, the effect of the good cluster points would not reduce with increasing power as the fuzzy memberships of these points in their respective clusters will be close to 1.

2. *Heuristic Approach*: This approach involves apriori removal of noise. If we have the knowledge of the distribution of the noise in the data set, then it could be removed prior to the cluster analysis. This is called the heuristic approach.
3. *Using Robust Techniques*: Some robust statistical techniques have been integrated into cluster analysis to give robust techniques of cluster analysis. Some of the examples include using the Least Median square (LMS) instead of Least Mean square. The breakdown point for this method is 0.5. i.e. The algorithm can correctly identify the clusters till the percentage of noise points is 50%. There are many other robust techniques for handling noise but all are computationally expensive as compared to the least squared approach. The study of these methods is outside the scope of this work.

The technique proposed by Dave [11] is an effective means of handling noise that is evenly distributed in a data set. This method is computationally less expensive than the robust statistics methods and has shown very good results for most of the data sets.

The performance of the squared error type algorithms is highly susceptible to noise, because each of the points including the noise points have to be assigned to one of the clusters. Hence, some of the noise points are assigned to the good clusters thus deteriorating the overall performance of the algorithm. To avoid classification of noise points into good clusters Dave[11] introduced the idea of a noise cluster. This is a cluster, in addition to the good clusters, in which all the noise points are dumped, thus making it unnecessary to classify them into good clusters. This method involves defining a similarity measure for the noise cluster. The similarity measure will decide the membership of each of the points in the noise cluster. To decide the similarity measure an entity called noise prototype has been defined. *Noise prototype* is a universal entity such that it is always at the same distance from every point in the data set.

Let v_c be the noise prototype, x_k be the point in the feature space, $v_c, x_k \in R^p$. Then the noise prototype is such that the distance of d_{ck} of point x_k from v_c is

$$d_{ck} = \delta \quad \forall k \tag{2.5.1}$$

If there are $(c\text{-}1)$ clusters in the data set, then let the c^{th} cluster be the noise cluster. For a specified value of δ, the minimization of the objective functional remains unchanged. Only the modified distance value has to be considered for the c^{th} cluster. The problem of selection of δ is very critical. If the value of δ is small, most of the points are classified as noise points. While if it is too large then most of the points are classified into the clusters other than the noise cluster. The average distance between the good points and their respective cluster centers is a reasonable approximation to the spread of the cluster. Hence the following formula for δ was proposed.

$$\delta = \lambda \frac{\sum_{i=1}^{c-1}\sum_{k=1}^{n}(d_{ik})^2}{(c-1)n} \tag{2.5.2}$$

where, λ is a multiplier used to obtain δ from the average distances.

The modified AFC algorithm based to be used for noisy data sets is called Noise Adaptive Fuzzy c-Elliptotype clustering algorithm (NAFC).

Algorithm 3 (NAFC)

1. Fix c the number of clusters, $2 \le c \le n$, where n is the number of data points; $c\text{-}1$ is the number of good clusters.

   Fix fuzzifier $m \in [1,\infty)$

   Initialize the value of mixing coefficients $\alpha_{i1}, \alpha_{i2} \in ((0,1), 1 \le i \le c)$

   Initialize membership matrix $U^{(0)}$

For iteration numbers I, I = 0,1,2...

3. Calculate the c fuzzy scatter matrices $\{S_{fi}\}^{(I)}$ from equation (2.4.8)
   Calculate the eigenvalues and the corresponding mixing coefficient. Calculate the value of δ from equation (2.5.2).
4. Update $U^{(I)}$ to $U^{(I+1)}$ using the equation (2.1.4) and the distance is measured by

$$(Z_{ik})^2 = \alpha_{i1} (D_{ik})^2 + \alpha_{i1} (D_{ik})^2 + (1-\alpha_i)(d_{ik})^2 \quad \text{for } i=1,\ldots,c-1 \qquad (2.5.3)$$
$$(Z_{ik})^2 = \delta^2 \quad \text{for } i=c$$

5. If $(U^{(I+1)}-U^{(I)}) \leq \varepsilon$ stop, else return to 2 with I=I+1.

Range image of the jig shown in Figures 2.1 to 2.3 with added noise has been shown below. NAFC algorithm has been used to segment the range image.

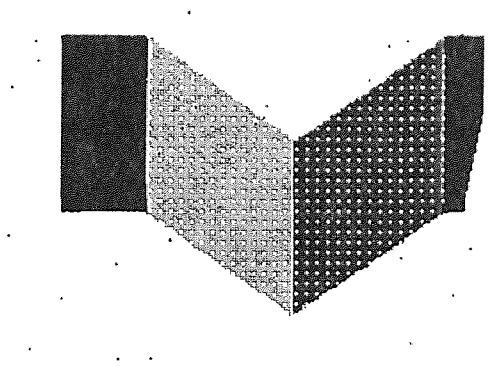


Figure 2.5 Range image of a jig with added noise segmented by NAFC
n =21149, c = 3, ε = 0.001

If instead of 3, the number of clusters given by the user is 4, the algorithm does not pick the two horizontal planes as separate planes. However to a human eye, these look separate. To make the algorithm pick these planes as separate entities the FCM component in the distance measure is increased by a small value of 0.001. The new distance measure is defined as

$$(Z_{ik})^2 = \alpha_{i1} (D_{ik})^2 + \alpha_{i2} (D_{ik})^2 + (1.001-\alpha_{i1}-\alpha_{i2})(d_{ik})^2 \qquad (2.5.4)$$

Noise distance measure, however, remains unchanged. The result obtained has been shown in Figure 2.6.

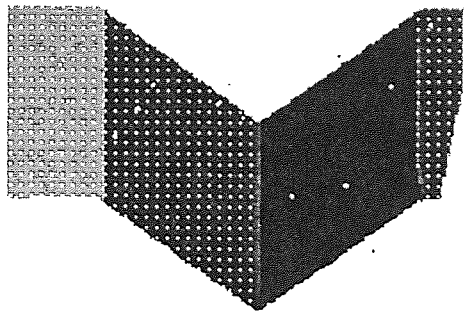


Figure 2.6 Range image of a jig with added noise segmented by NAFC (with a higher FCM component)
$n = 21149$, $c = 3$, $\varepsilon = 0.001$

2.6 Problem of Unknown Number of Clusters

The problem of unknown number of clusters has been a difficult problem to solve. Dave [12] observed that even if the number of clusters were over-specified the representation of boundaries for boundary detection was acceptable. Krishnapuram and Freg [20] came up with a concept of cluster merging. However, none of these techniques were found to be effective. Recently Krishnapuram and Frigui[14] introduced the idea of competitive agglomeration(CA).

The CA algorithm starts by specifying a large number of clusters. As the algorithm, progresses the adjacent clusters compete for data points. Some clusters lose the competition, shrink and finally vanish. As the iteration proceeds, we get a partitioned data set with progressively diminishing number of clusters. When the objective functional reaches the global minimum the data set is partitioned into optimal number of clusters. This method is quite insensitive to initialization and to local minima. It starts with highly over-specified number of clusters and hence it does not get affected by initialization.

In this case the objective function to minimize is:

$$J = \sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ik})^2 d^2 - \psi \sum_{i=1}^{c}\left[\sum_{k=1}^{n} u_{ik}\right]^2 \tag{2.6.1}$$

subject to

$$\sum_{i=1}^{c} u_{ik} = 1 \quad \text{for } k = 1,\ldots,N \tag{2.6.2}$$

The first term is the FCM objective function as in (2.1.1). It allows us to control the size and the shape of the cluster. The second term controls the number of clusters. The number of clusters are dynamically updated. The minima of the sum of intra-cluster distances reached while partitioning the data into minimum number of clusters.

The cardinalitity of a cluster is defined as

$$N_s = \sum_{k=1}^{n} u_{sk} \tag{2.6.3}$$

The minimization of the objective function gives

$$u_{ij} = u_{ij}^{\text{FCM}} + u_{ij}^{\text{BIAS}} \tag{2.6.4}$$

where,

$$u_{ik}{}^{FCM} = \frac{1/d_{ik}{}^2}{\sum_{j=1}^{N}\left(1/d_{jk}{}^2\right)} \tag{2.6.5}$$

$$u_{ik}{}^{Bias} = \frac{\alpha}{d_{ik}{}^2}\left(N_s - \overline{N}_t\right) \tag{2.6.6}$$

$$\overline{N}_t = \frac{\sum_{i=1}^{c}\frac{1}{d_{ik}{}^2} N_i}{\sum_{i=1}^{c}\frac{1}{d_{ik}{}^2}} \tag{2.6.7}$$

N_S is the cardinality of the cluster and $\overline{N}_t$ is the weighted average cardinalities of the cluster from the point of view of point x_t. The weighting factor is inversely proportional to the square distance of the cluster center from the point. For the clusters of high cardinality the bias term is positive. Hence, the memberships of the points in this cluster increase . When the cardinalities of the cluster is below average, the bias term becomes negative and hence the membership of the point with respect to the cluster decreases. Therefore, some of the clusters become larger while others go on diminishing in size. When the memberships of the points in a given cluster go on increasing, its cardinality increases and vice-versa. When the cardinality of the cluster drops below a certain threshold, the cluster is rejected. At the end, only a few clusters will survive and others will shrink and eventually be extinct.

If a point is very close to a cluster then the weighting factor is large and $N_S \cong \overline{N}_t$. Therefore the biasing term in the equation (2.6.6) is 0. Thus the term u_{ij} is equal to u_{ij}^{FCM} and the choice of α is very crucial. Here, α is given by the following formula

$$\alpha = \eta \frac{\sum_{i=1}^{c}\sum_{k=1}^{N}(u_{ik})^2 d_{ik}^{\,2}}{\sum_{i=1}^{c}\left[\sum_{k=1}^{N} u_{ik}\right]^2} \qquad (2.6.8)$$

Here η exponentially decreases with the iteration number, thus trying to reduce the effect of agglomeration as the algorithm progresses.

The algorithm mentioned above works very well even with the AFC distance measure and fuzzifier m=2. But with noisy range images agglomerated NAFC runs into problems.

In noise clustering approach, it is suggested that the noise cluster should not be eliminated , however small its cardinality might be. The initial noise distance may be very small. When the algorithm starts, it has a tendency to agglomerate the data points very fast as the value of η is large. Due to this the algorithm tends to classify all the

points into one noise cluster and then reject the other clusters as there cardinalities are below a certain threshold.

To tackle this problem an adaptive approach is suggested. Let the threshold be variable instead of being fixed. Let the new threshold (T) approach the old threshold value asymptotically with respect to the number of iterations. The new threshold is defined as

$$T = T_0\left(1 - e^{-k/\tau}\right) \tag{2.6.9}$$

Here, T_0 is the threshold value at a very large iteration number. τ is the time constant k is the iteration number.

Here, the threshold for elimination of the cluster starts with 0. Hence it is possible to agglomerate small clusters when the noise distance is very small. As the noise distance increases and the cluster grows, the threshold value also increases. When the good clusters compete with each other for data points, some of them grow while other clusters shrink and go below T_k and are rejected.

Algorithm 4

1. Fix c the number of clusters, $2 \le c \le n$, where n is the number of data points; c-1 is the number of good clusters.

   Fix fuzzifier $m \in [1,\infty)$

   Initialize the value of mixing coefficients $\alpha_i \in ((0,1),\ 1 \le i \le c)$

   Initialize the value of η_0, T_0.

   Initialize membership matrix $U^{(0)}$

For iteration numbers I, I = 0,1,2...

2. Calculate the c fuzzy centers $\{v_i\}^I$ using memberships $U^{(I)}$ from equation (2.1.3).
3. Calculate the c fuzzy scatter matrices $\{S_{fi}\}^{(I)}$ from equation (2.4.8)

   Calculate the eigenvalues and the corresponding mixing coefficients. Calculate the value of δ from equation (2.5.2).

4. Calculate the value of η_i and T_i .
5. Calculate the cardinalities using (2.6.3). Eliminate the cluster if cardinality is less than T_i
6. Update $U^{(I)}$ to $U^{(I+1)}$ using the equation (2.6.4) and the distance is measured by equation (2.5.3).
7. If $(U^{(I+1)}-U^{(I)}) \leq \varepsilon$ stop, else return to 2 with I=I+1.

As shown in Fig. 2.6, this approach is known to show very good results.

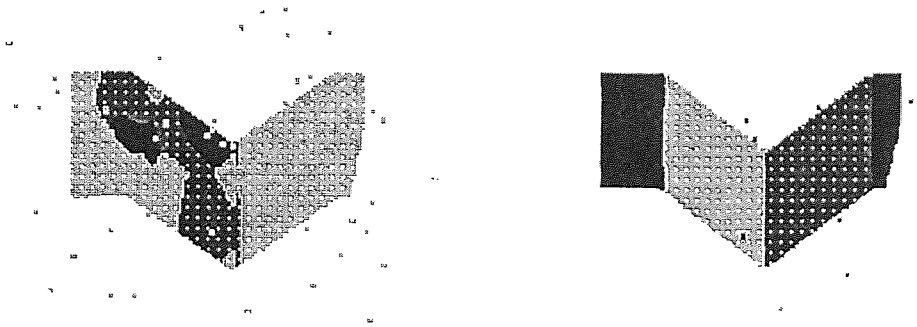


Figure 2.7 Intermediate and Final Stages in Competitive Agglomerated Clustering of a Range Image
$n = 21149$, $\varepsilon = 0.001$, Starting $c = 12$, Final $c = 3$

2.7 Random Sampling

Cluster analysis methods discussed in this chapter are computation intensive. If the number of points in the range image is large then it takes a long time to analyze the data. Fuzzy clustering techniques suffer from this drawback. Random sampling technique has been proposed to reduce the computation time.

Method of random sampling in cluster analysis involves selecting a sample m points randomly from a data set of n points. The number m be as small as possible to reduce the number of computations, but should be sufficiently large so that it gives a realistic approximation of the original data set. One of the fuzzy clustering algorithms is then used to segment the smaller data set. As the number of data points reduce the number of computations and hence the computation time reduces drastically. When

maximum change in fuzzy memberships of the points reaches below certain value ε, all the points in the cluster are invoked. By this time a good estimate of cluster prototypes is obtained. A few more iterations with all the data points gives completely segmented data. The procedure decreases the computation time drastically for very large data sets.

However, this method does not work when NAFC algorithm is used with small number of noise points. In this case the probability of choosing the noise point in the randomly chosen points is very less. Algorithm tries to fit (c+1) number of clusters in a data set without noise where c is the number of clusters specified by the user. This gives a wrong estimate of the cluster prototypes and the algorithm takes a long time to converge.

CHAPTER 3
PLANE FITTING

3.1 Introduction

In chapter 2, fuzzy clustering algorithms were discussed in detail. Fuzzy membership of a point in a cluster is a measure of relative proximity to a cluster prototype. A point is classified into a cluster in which it has a highest membership. The data set, thus obtained, after classification is crisp or hard partitioned, i.e. each point can belong to one and only one of the clusters. Each of these clusters, represent the faces of object in the range image.

Geometric properties of object in the range image cannot be obtained without the knowledge of properties of its bounding planes [27]. Properties of finite planes that form the bounding planes are not known, since the fuzzy partitioned data obtained from the clustering algorithm is in the form of distinct points in three dimensional space. These points could be hard partitioned as mentioned above. (Refer to Figures 2.1-2.5) However, relationship needs to be established between the points in a cluster. Hence, the partitioned data set obtained from clustering algorithm needs to be analyzed further, to get the information about the planar features in the range image.

Properties of planar cluster, like the plane equation, are determined from the position of points that belong only to that cluster. It is possible to eliminate undesirable effect of points in noise and other clusters by considering only those points that have a membership in that cluster above a certain value α. This ensures that the data point under consideration does not lie on any other plane. The value of α is chosen heuristically. The data set is said to be *α-cut hard partitioned*. Although, α-cut hard partitioning has not been used in the present problem, it may be used to improve the performance of plane identification algorithms.

A finite plane in a 3-D space is completely defined by [1]:

1. The equation of an infinite plane of which the finite plane is a subset.

2. Bounding curve which is coplanar to the infinite plane defined by the above equation.

Equation of a plane can be derived from the equation of the coplanar curve. But, generally it is explicitly referred to while defining a plane [1].

This chapter will deal with the problem of finding the equation of a plane represented in by a cluster in a fuzzy partitioned range image. It is called the problem of Plane Fitting [19].

The first method of plane fitting is the *Least Mean Squared Approach.* It is the most popular method of plane fitting. If an equation of a plane passing through any number of points greater than three has to be obtained, then this method gives a very good fit. Least square fit can also be used for curve and surface fitting of various shapes with minor modifications. However, it suffers from certain drawbacks.

To overcome the drawbacks of the least square fit method alternative approach called the *Eigenvector Approach* is discussed. It has certain advantages over least mean squared approach. There is an additional advantage while using this method with fuzzy clustering algorithms. Both the algorithms have been discussed in detail in the following sections.

3.2 Least Mean Squared Approach to Plane Fitting

The least square approach was developed by Gauss [19] to fit lines through a given set of points in 2-D. Here, we use a similar method to fit a plane though points in a 3-D space. This method is also called regression analysis. In regression analysis, two of the three variables of point P[x,y,z] are considered as independent variables while the third is the dependent variable. In this case we consider z as the independent variable so that the equation of a plane will be of the form $z=f(x,y)$. An attempt is made to find unique function f such that all the points P lie on the entity defined by the function. But when the points are scattered a unique function does not exist. So, a mean $f(x,y)$ of z has to be found to approximate the plane. The function $f(x,y)$ is called the regression

function. In this case, the function should be linear as planar entities have to be identified.

The plane should be fit through the given points such that the sum of the squares of the distances of those points from the plane is minimum, where the distance is measured in vertical direction (i.e. the z direction)[19].

Consider a sample (x_1,y_1,z_1), (x_2,y_2,z_2),... (x_n,y_n,z_n) of n points. The equation of the plane in the form $z=f(x,y)$ is $z=a+bx+cy$. The vertical distance of a point j from the plane is $[z_j-a-bx_j-cy_j]$. Hence the sum of the squares of these distances is

$$q = \sum_{j=1}^{n} \left[z_j - a - bx_j - cy_j \right]^2 \tag{3.2.1}$$

In the method of least squares we choose a, b, c such that the value of q is minimized. q depends on a, b and c and a necessary condition for q to be minimum is

$$\partial q/\partial a = 0 \qquad \partial q/\partial b = 0 \qquad \partial q/\partial c = 0 \tag{3.2.2}$$

$$\partial q/\partial a = -2\Sigma(z_j-a-bx_j-cy_j) = 0 \tag{3.2.3}$$

$$\partial q/\partial b = -2\Sigma x_j (z_j-a-bx_j-cy_j) = 0 \tag{3.2.4}$$

$$\partial q/\partial c = -2\Sigma y_j (z_j-a-bx_j-cy_j) = 0 \tag{3.2.5}$$

The values of a, b, c can be obtained by solving the matrix equation

$$\begin{bmatrix} n & \sum x_j & \sum y_j \\ \sum x_j & \sum x_j^2 & \sum x_j y_j \\ \sum y_j & \sum x_j y_j & \sum y_j^2 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} \sum z_j \\ \sum x_j z_j \\ \sum y_j z_j \end{bmatrix} \tag{3.2.6}$$

Simplifying, $z_j=a+bx_j+cy_j$, equation the form $a'x+b'y+c'z+d' = 0$, can be obtained.

Drawbacks:

1. For the planes in the range image parallel to Z axis, the one to one relation between the z co-ordinate and the (x,y) pair ceases to exist. So, the plane can no more be defined as $z=a+bx+cy$. The least square fit approach fails in such a case.
2. Least square approach is extremely susceptible to noise. It has a break down point 0. i.e. even a single noise point can affect the result adversely.

Noise is not a problem, while using least square fit after segmenting the image using NAFC. Noise has already been eliminated in NAFC algorithm. However, the first problem of vertical planes is often encountered. Hence a better plane fitting method needs to be devised.

Eigenvector approach overcomes most of the drawbacks of the least mean squared approach to plane fitting.

3.3 Eigenvector Approach to Plane Fitting

Eigenvector approach to plane fitting uses the information about the distribution of points around the centroid to compute the equation of a plane: This algorithm uses a scatter matrix to estimate the distribution of points.

Scatter matrix of n points in 3-D space can be defined as:

$$S = \sum_{k=1}^{n} (x_k - v)(x_k - v)^T \qquad (3.3.1)$$

where,

x_k is the position vector of k^{th} point in 3-D space.

v is the postion vector of the centroid.

Scatter matrix contains all the information about the geometry of a cluster. The three eigenvectors are directed along the three principal axis of an ellipsoidal cluster. The eigenvalues of the scatter matrix indicate the type of cluster and a spherical and a planar cluster are special cases of ellipsoidal clusters. For a spherical cluster, the three eigenvalues of the cluster will be nearly equal. This indicates that the distribution of

the points in the three orthogonal directions is equal. In case of planar cluster, one of the eigenvalues is very small as compared to the other two. The eigenvectors corresponding to the two larger eigenvalues defines the plane passing through the cluster and the third eigenvector is normal to the plane.

Let λ_1, λ_2, λ_3 be the eigenvalues of the scatter matrix and u_1, u_2, u_3 be the three corresponding eigenvectors. The plane is defined by the two larger vectors among $\lambda_1 u_1$, $\lambda_2 u_2$, $\lambda_3 u_3$. The plane passes through the centroid v_i. A plane passing through a point and two non-collinear vectors can be determined as follows.

The equation of the plane is in the form $ax+by+cz+d=0$. Any point P(x,y,z) lying on the plane must satisfy the equation. Consider a position vector $\underline{p}$ of any arbitrary point in the plane and $\underline{v}$ the position vector of the cluster center. Then the vector $(\underline{p}\text{-}\underline{v})$ lies in the plane of u_1 and u_2.

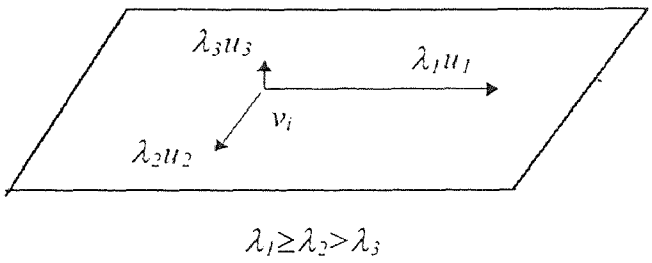


Figure 3.1 Definition of a plane using eigenvectors of scatter matrix

Hence the vector triple product or the scalar product

$$(\underline{p}\text{-}\underline{v}).(u_1 \times u_2) = 0 \quad \text{or} \quad (\underline{p}\text{-}\underline{v}).u_3 = 0 \tag{3.3.2}$$

Solving the above equation we get the following representation of the plane.

$$l_3 x + m_3 y + n_3 z - (l_3 x_i + m_3 y_i + n_3 z_i) = 0 \tag{3.3.3}$$

Eigenvector method of plane fitting is preferred to least squared approach as:

- The problem of vertical planes is eliminated by the Eigenvector method.
- This method can also give an estimate of the distribution of the data points about the plane.

When this method is used as an extension to AFC or GK clustering algorithms, it has an added advantage. These clustering methods use fuzzy scatter matrix to obtain a distance measure. Their eigenvalues and eigenvectors have also been computed. Centroid of the data set is the cluster center of the respective cluster.

Hence, equation 3.3.2 can be directly used to compute the equation of the plane. This saves some computations while calculating the plane equations.

This approach is extremely simple and less susceptible to errors. Also it is computationally efficient. It overcomes the drawback of the earlier method of least square fit and henceforth has been used to obtain the equations of planes.

Equations of planes have been obtained using the eigenvector approach with fuzzy partitioned data. The results have been combined with results of the algorithm for identification of bounding edges. These have been shown in Figures 4.4 to 4.9.

CHAPTER 4
DETECTION OF BOUNDING EDGES OF PLANES

4.1 Introduction

A plane equation obtained in section 3.3 represents an infinite plane. Orientation of plane in 3-D space can be calculated by using equation of a plane. Surface normals can be calculated from equation of a plane. It is an important parameter in many applications such as generating graphic displays or NC machining. If faces of an object are planes then the bounding edges are straight lines. The bounding edges form a closed loop that divides a plane into two parts- interior and exterior of a loop. However, we are restricting the analysis to objects that have no holes. So interior of a plane is the interior of a single loop. Information about a finite plane is incomplete unless information about the bounding edges is known [27].

To define a loop of straight lines, following features have to be defined:

1. *Equation of a Straight Line*: An equation defines orientation of a line in space. However, it represents an infinite line. Like planes, all edges of an object are finite. So, an additional constraint is necessary to define an edge.
2. *Two Endpoints of a Straight Line*: This constraint defines a line segment completely. If necessary, the equation of the line could be obtained from the two endpoints.
3. *Position of a Line in a Loop*: A loop is an ordered set of non intersecting curves in space. Hence the position of the individual lines in a loop is an important parameter in defining a line loop.

Since edges of an object are formed by intersection of its faces, equation of edges can be obtained using equations of the planes. However, the existence of such an edge is not always guaranteed. Since the faces of the object are finite, the planes may not intersect at all. Refer to Figure 4.1. An edge validity criterion has been developed to overcome this problem. Also, a method to find the endpoints of the edge has been developed. The edges are further linked together to define a loop.

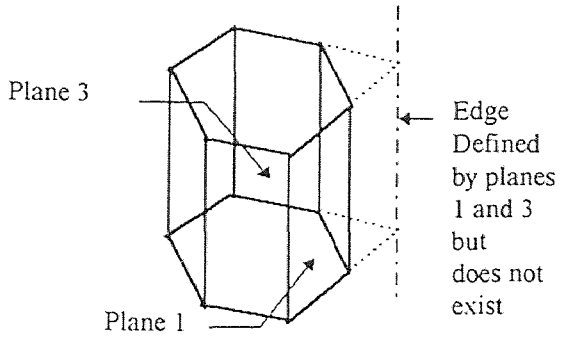


Figure 4.1 A schematic diagram to explain necessity of edge validation

4.2 Determining Equation of Edge Using Plane Intersection

To find an equation of a line in 3-D space, consider line as intersection of two planes. The equation of a plane in 3D Cartesian space is of the form $ax+by+cz+d=0$. Consider two planes

$$a_1 x+b_1 y+c_1 z+d_1=0 \qquad (4.2.1)$$
$$a_2 x+b_2 y+c_2 z+d_2=0$$

Equation of a line passing through a point (x', y', z') in a 3D Cartesian space is of the form:

$$(x-x')/l = (y-y')/m = (z-z')/n = k \qquad given\ l, m, n \neq 0 \qquad (4.2.2)$$

This equation of a line shows that each of the variable x, y, and z can be expressed in terms of the parameter k. The equation of the line can also be represented in the form

$$y = f_{1y} + f_{2y}\, x \quad \& \quad z = f_{1z} + f_{2z}\, x \qquad (4.2.3)$$

Solving for the equation of the line

$$f_{1y} = (d_2 c_1 - d_1 c_2)/(b_1 c_2 - b_2 c_1) \qquad f_{2y} = (a_2 c_1 - a_1 c_2)/(b_1 c_2 - b_2 c_1)$$
$$f_{1z} = (d_2 b_1 - d_1 b_2)/(b_2 c_1 - b_1 c_2) \qquad f_{2z} = (a_2 b_1 - a_1 b_2)/(b_2 c_1 - b_1 c_2) \qquad (4.2.4)$$

Equation of a line thus obtained may represent an edge of a body when properly constrained. A procedure has been developed to validate the existence of an edge.

4.3 The Edge Validity Criterion

A procedure has been developed for checking the existence of an edge in a body represented by the range image. This procedure is called the *edge validity criterion*.

The procedure takes a heuristic approach while checking whether the edge is seen in the range image. It is easy to comprehend that if the two faces of the body do not intersect to form an edge of a body, then no points on the surface of a body would lie on the edge. Conversely, if the edge formed by the intersection of two faces exists, then there should be at least some points on each of the faces that lie on the edge. The edge validity criterion uses a similar argument to check the validity of an edge in a range image.

Consider an edge of a body obtained by the intersection of two planes. The equation of this line represents an infinite line. There are two methods of determining whether the points belonging to a particular cluster in the segmented range image lie on the line depending on:

1. The position of the points.
2. The fuzzy memberships of the points.

The first method involves finding the number of points lying on the edge by their position. However, due to the errors in the measuring device and the errors in the computation the points may not exactly lie on the edge. Hence, a following approach is proposed.

Consider a cylindrical volume of radius ε and infinite length around the edge under consideration. Here, ε is the expected error in determining the point position. So,

all the points that lie inside the cylindrical volume may be assumed to belong to the edge.

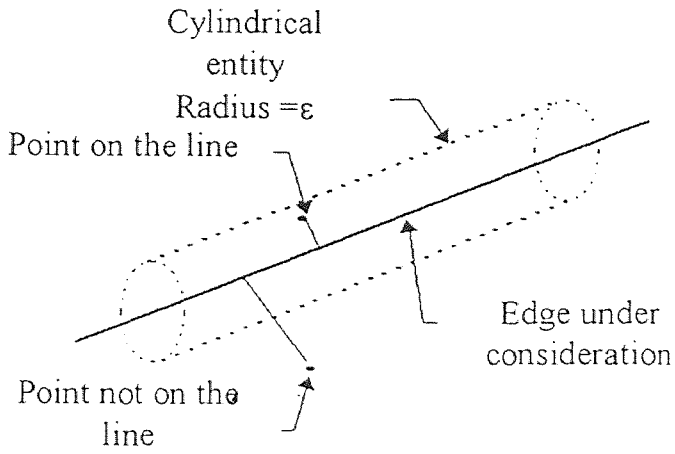


Figure 4.2 Cylindrical volume of interest for the edge criterion.

As discussed earlier, the two planes intersect to form an edge. To determine the validity of such an edge, all the points that belong to the two planes are considered. If the number of points that lie inside the cylindrical volume is larger than a certain threshold then the edge is valid. Otherwise the edge does not exist.

The decision of ε and the threshold is critical. It is generally made through experience and trial and error. The value of ε could be guessed as average of the inter-point distance in the two clusters. The threshold is some percentage of the points in the each of the clusters (say 5%).

The second approach for the edge validity criterion is the fuzzy membership approach. If the fuzzy memberships of a point in the two planar clusters defining the edge are comparable, then the point may be said to belong to both the planes.

Let u_i and u_j be the two highest fuzzy memberships of a point that belongs to cluster I such that $u_i \geq u_j$. The point is said to be on the edge defined by the two planes i and j if

$r \leq (u_j / u_i) \leq 1$ where $r \cong 0.8$.

The number of such points in both the clusters are counted. If the number of such points in the cluster is higher than the threshold then the edge exists. The fuzzy memberships are already available, so this method saves extra computations. It is found to be equally effective as the first method.

Edge defined by planes 1 and 3 in Figure 4.1 has been rejected while constructing the hexagonal prism. The result has been shown in Figure 4.5.

The edge validated by one of these two methods is known to exist. The next task is to limit the edge. The two extreme points of the edge have to be determined.

4.4 Edge Delimitation

Once a valid edge is determined, the next task is to determine the endpoints of the edge so that it can be restricted. These endpoints are the vertices of the solid. For a solid with planar faces the vertex is defined as the point of intersection of three or more faces. A vertex has at least three edges merging into it. So a vertex forms a delimiter for at least three edges. Number of vertices in a body can be determined from the Euler's formula [23, 27]:

$$F - E + V - L = 2(B - G) \qquad (4.4.1)$$

where,

F = Number of faces

V = Number of vertices

E = Number of valid edges

L = Number of faces' inner loop

B = Number of bodies

G = Number of genus (through holes)

Since the scope of work is restricted to objects without holes we always consider L and G to be 0.

Euler's equation gives an estimate of number of vertices. This is calculated after execution of the previous program and is not known at the time of compiling the

program. In case of efficient memory management was essential, the total number of vertices needed could be computed by Euler's equation. Memory could then be dynamically allocated. Hence, Euler's equation facilitates efficient storage of vertices. Endpoints of an edge would then be specified by a reference or a pointer to the vertices of the solid object.

Endpoints of edge could also be explicitly defined for each edge. This approach is preferred to the prior because of simplicity in data access. This is discussed in section 5.2.

The vertices are points in 3D Cartesian space. Let a vertex V(x,y,z) be the point of intersection of three planes whose equations are:

$$\begin{aligned} a_1x+b_1y+c_1z+d_1=0 \\ a_2x+b_2y+c_2z+d_2=0 \\ a_3x+b_3y+c_3z+d_3=0 \end{aligned} \tag{4.4.2}$$

The co-ordinates x, y, z could be obtained by solving the following matrix equation.

$$\begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} -d_1 \\ -d_2 \\ -d_3 \end{bmatrix} \tag{4.4.3}$$

If the matrix is singular then two or more planes are parallel and three planes do not intersect in a single point [19]. Such combinations, that make the coefficient matrix singular or nearly singular, are rejected.

The points obtained this way may not necessarily represent a vertex. In the actual object, the planes considered in the above equation may not intersect (i.e. may terminate before intersecting). Therefore, a technique is proposed to choose the appropriate vertex points.

Referring to Figure 4.2 all points inside the cylindrical volume are assumed to be on the edge. For analysis, the points that belong to two clusters and lie inside the cylindrical volume are considered. Projection of each of these points on the edge gives a corrected position of these points (i.e. the position of the points assuming zero error). Projections of the points are found by drawing normal on the edge as shown in the Figure 4.2. Extreme projections of points belonging to each of the two clusters defining the edge are considered. Extreme projections are the maximum and the minimum values of any co-ordinate (x, y or z) along the edge. Thus we have one maximum and one minimum projection for each of the two clusters. Two projections that correspond to the lower maximum value and the higher minimum value are taken as the approximate endpoints of the edge. Thus a shortest possible edge is chosen. This in turn means, that only the region in which both the clusters have some points lying on the edge is chosen.

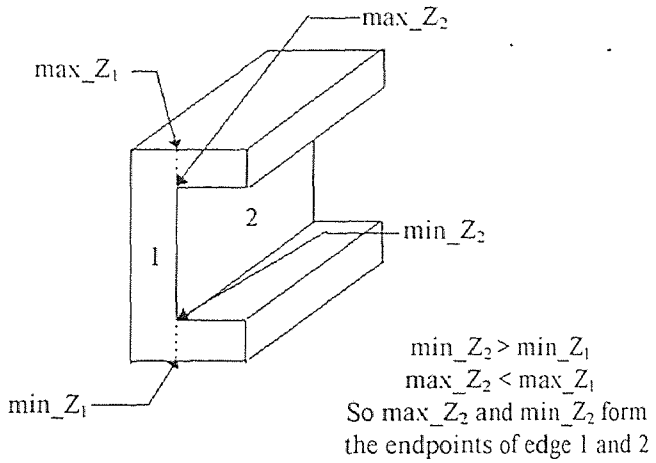


Figure 4.3 Example of edge delimitation

But, since the range image consists of discrete points this estimate is not a very good estimate. Besides, this approach does not guarantee the loop closure. The result can be improved by using the knowledge of the vertices.

To find the endpoints of an edge formed by the intersection of planes i and j, the vertices defined by the planes i, j, and k (where k=1,....c; $k \neq i$, $k \neq j$ and c is the number of planes in the object) are considered. The vertex nearest to each of the endpoints determined by the earlier method is the actual vertex and the endpoint of the edge. This method serves a dual purpose:

- Validating the existence of the vertex.
- Improving the estimation of the edge endpoint.

This estimation of the edge endpoints is found to work very well as can be seen from the results shown in Figure 4.4 - 4.9.

The two endpoints of an edge defines the edge completely. Thus the discrete edge entities have been completely defined.

4.5 Edge Connecting and Plane Bounding

The edges obtained from the method in section 4.4 are discrete linear entities. A closed loop of co-planar edges along with the equation of the infinite plane defines a closed planar entity completely. The edges that lie a plane need to be connected in an appropriate order to obtain a closed loop which forms the bounding entity [23]. The procedure for sorting the edges for a particular plane and arranging them in proper order goes as follows:

1. Start with any edge. Consider any extreme point on the edge as the start point. Then the other end of the edge is the endpoint.
2. Search for the next edge that has one of the extreme point same as or close to the endpoint of the previous edge. The allowable error could be decided apriori depending on the distribution of the points in the range image. However, generally the error is 0 because the points are derived from the same vertex entities.

   Call this point as the start point and the other point of the edge as the endpoint.

   If such a line is not found then the task is unsuccessful and the loop obtained is not a closed loop.

3. Check whether the endpoint of the present edge is same as or close to the start point of the first edge. If yes, then report success.

   Otherwise, go to step 2.

Note that once an edge is selected, it is removed from the available set for that plane thus avoiding repetition of the edges.

The loop obtained from the above algorithm is a closed loop. This along with the equation of the plane identifies the face of an object completely. This information is then formatted to store it in a data structure used to generate a solid model.

The scope of this thesis was restricted only to objects without holes. Figures 4.4 - 4.9 show that the algorithm works very well for range images of such objects.

However, in case of objects with holes, the algorithm runs into difficulties due to the following reasons:

1. A body with a hole may have two closed loops for a single plane equation. In this algorithm one plane equation has been associated with only one closed loop. Hence the algorithm may not work in such cases.

Further work is necessary to incorporate multiple loops with one plane equation.

Results obtained by using the procedure explained in sections 3.3 and 4.2 to 4.4 has been used to construct wire frame models of parts. These models have been shown in Figures 4.4 to 4.9.

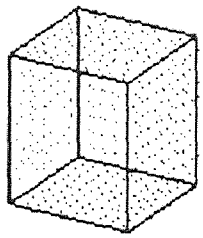


Figure 4.4 Wire frame model of a cube. Range image is segmented using NAFC and edges detected using the algorithm suggested $n = 622$ $c = 6$ $\varepsilon = 0.001$

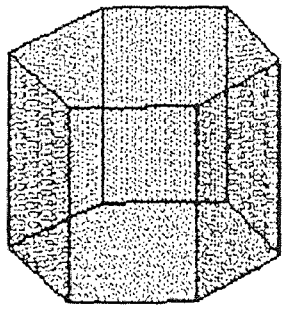


Figure 4.5 Wire frame model of a hexagonal prism $n = 8400$ $c = 8$ $\varepsilon = 0.001$

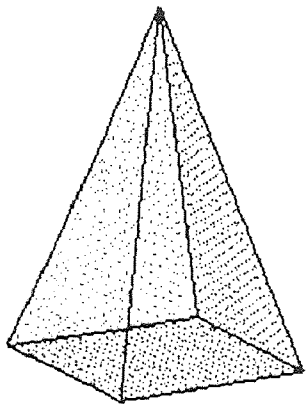


Figure 4.6 Wire frame model of a square pyramid
$n = 1983 \quad c = 5 \quad \varepsilon = 0.001$

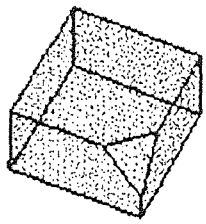


Figure 4.7 Wire frame model of a cube with chamfered corner
$n = 1628 \quad c = 7 \quad \varepsilon = 0.001$

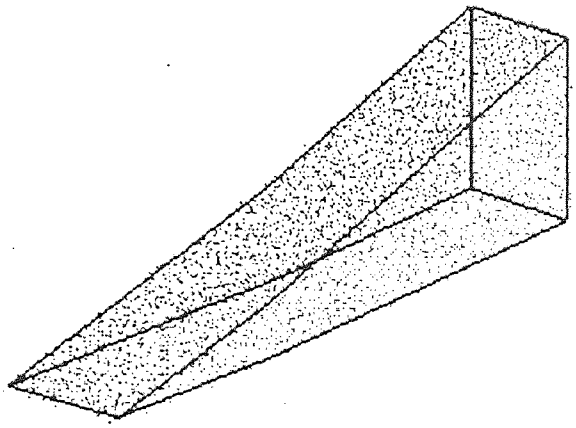


Figure 4.8 Wire frame model of a triangular wedge
$n = 4428 \quad c = 5 \quad \varepsilon = 0.001$

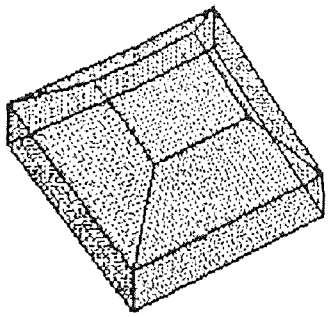


Figure 4.9 Wire frame model of a paper weight
$n = 5072 \quad c = 10 \quad \varepsilon = 0.001$

CHAPTER 5
SOLID MODEL CONSTRUCTION

5.1 Introduction

Geometric model of a part is its informationally complete mathematical representation. Solid modeling which is an important aspect of geometric modeling is used to create and communicate the shape information of an object. A solid model may have volumetric and physical information about the object in addition to the topological properties of the object. As discussed earlier in section 1.3 there are 6 main types of solid modeling techniques. But, only three of these are predominant[23, 26, 27].

1. The *Constructive Solid Geometry* (CSG)
2. The *Sweep Representation*
3. The *Boundary Representation* (B-Rep)

Range image of an object gives information about the surface of an object. So it is easier to construct boundary representation of the object. B-Rep method of object representation is easier for porting across various systems. Neutral files like the Initial Graphics Exchange Specification (IGES) use the boundary information to exchange data. Taking the above two factors into consideration, the object is modeled by boundary representation.

5.2 Graph Based Models

A geometric model emphasizing the topological structure, with data pointers linking together an object's faces, edges and vertices is a *graph based model* [26]. A solid object can be represented as a list of object's faces and their respective surface equations. The edges of these faces are represented as curve equations with pointers to endpoints. If each of the edge and the vertex occurs only once in a data structure then the entities are connected by pointers or connectivity matrices. However, with the problem at hand, the bounding edges and the bounding vertices have been explicitly defined. So, the problem of connectivity does not arise. The model, thus generated, is a

tree based model which is a specialized case of a graph based model. It contains a lot of redundant information, but in many cases, such information is useful in further processing.

Tree is a special type of graph in which each of the node, except one, has one and only one parent but may have any number of children. Root is the node in a tree that has no parent. Leaf is a node that has no child. A tree has only one root and more than one leaves. There is a unique path from the root to any given node in the tree.

These properties of a tree graph allow easy access of the data. If each of the node represents an entity then the properties of the parent entity can be easily known as there is a single parent. The access to the child entities is facilitated by the pointers.

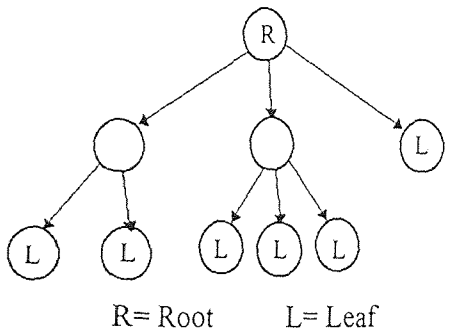


Figure 5.1 Typical representation of a tree

Boundary representation of an object is infact a graph based model. In B-Rep model, the node represents an entity.

5.3 Boundary Representation (B-Rep) Model

As discussed in the previous section, boundary representation of an object is a graph based representation [26]. However, it has some additional features.

A general data structure for a boundary representation should have both topological and geometric information [27].

A data structure shown in Fig. 5.2 is based on equation 4.4.1.

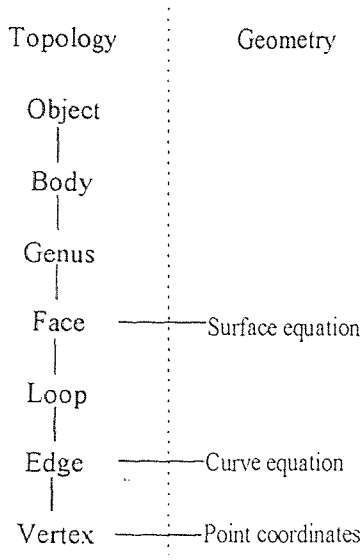
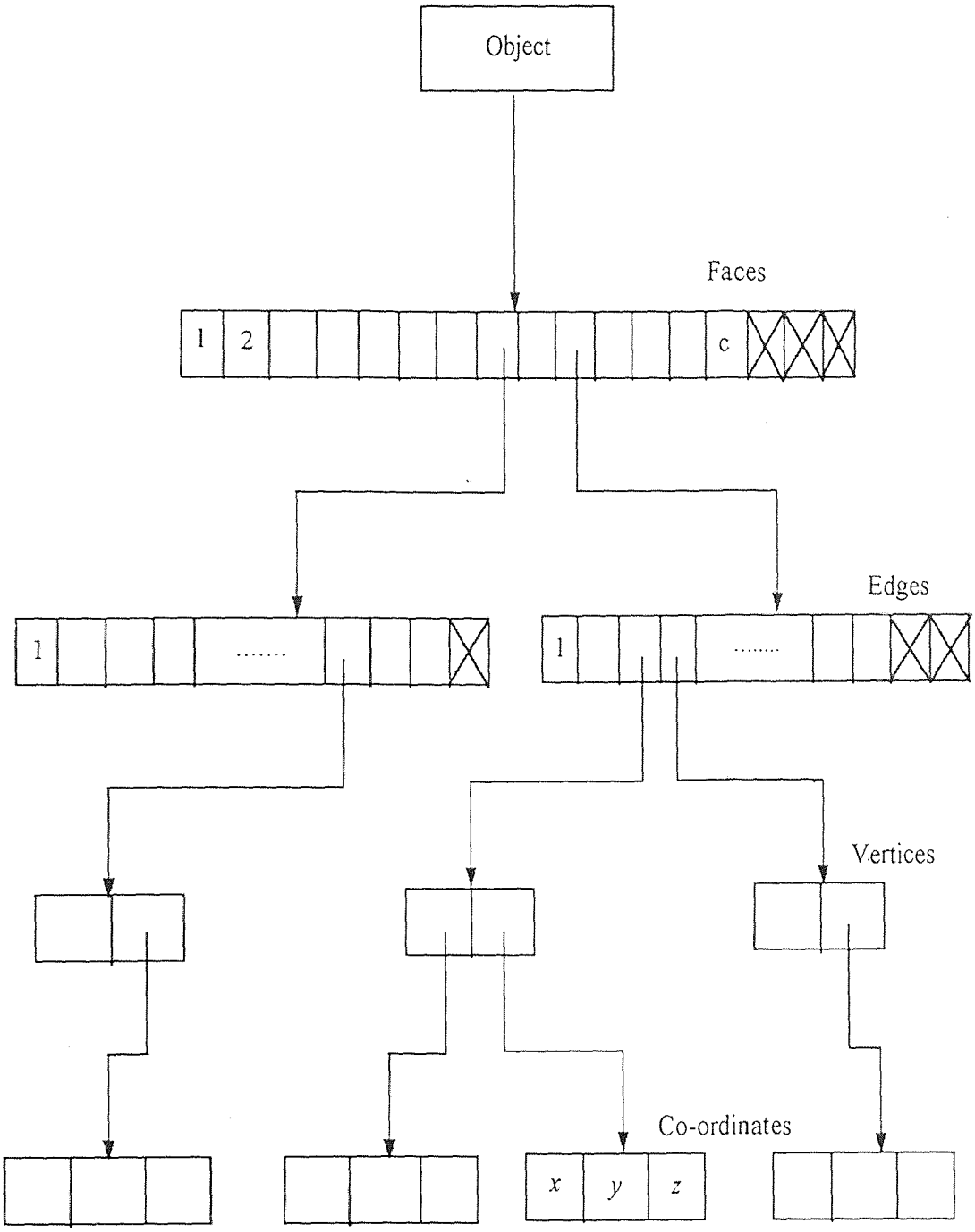


Figure 5.2 General data structure for boundary modeling

The bounding surface should be well formed (i.e. it should be closed form, orientable, non-self -intersecting, bounding and connected). To satisfy these criteria the faces must have the following properties. [23]

1. A finite number of faces define the boundary.
2. A face of a solid is subset of solid's boundary.
3. The union of all faces defines its boundary.
4. A face is itself a subset or limited region of some more extensive surface.
5. A face must have a finite area and must be dimensionally homogeneous.

To generate a solid model with the above properties a modified tree has been designed to suit our requirements. It is shown in Fig. 5.3. The maximum number of faces that an object can have is equal to the maximum number of clusters that the algorithm can handle. The crosses in the cells of the tree indicate that the positions are unoccupied. The unoccupied position represents a node that does not have any children. The maximum number of edges is decided apriori. The information about the



Crosses indicate that the location is empty

Figure 5.3 Boundary Representation tree for an object with planer faces

number of bounding edges for a face is stored in a face node along with the equation of the faces. The edges, for a given face, are ordered such that a loop is traced as we go from left to right in the array. Edge endpoints are described by 3 co-ordinates.

The tree is implemented in the program by defining a following structures.

```
typedef struct{
        double coord[3];
} point; /* Structure for a point in 3-D space
typedef struct {
        point endpoint[2];
}line;
typedef struct {
        int no_borders; /* Number of bounding edges */
        Boolean closed; /* True or false whether the bounding entity is closed */
        double coeff[4]; /* Coefficients of  equation of the plane */}
        line edge[20]; /* Assumption max. number of edges for a plane=20 */
}surface;
typedef struct {
        int no_faces; /* This stands for the number of faces in a solid */
        surface faces[MAX_CLUST]; /* MAX_CLUST is the max. number of
                                        clusters that clustering algo. can handle*/
} solid;
```

Figure 5.4 Tree Implementation in C

The data structure shown above stores all the relevant information. This information constitutes a solid model.

5.4 Data Transfer to CAD Systems

A solid model of the object generated as described in the previous section has to be transferred to a commercial CAD software to be useful for industrial applications. To transfer the data from the new system to a commercial CAD systems, translators can be developed for each of the CAD software. However it is impractical and impossible to develop such interpreter for each system. So standard neutral files are used to transfer data from the newly developed system to the CAD packages. IGES, PDES and many other file formats are available.

Initial Graphics Exchange Specification (IGES) establishes information structures to be used for representation and communication of product definition data. The use of this specification permits the compatible exchange of product data across Computer Aided Design and Computer Aided Manufacturing systems. The specification involves a file format, a language format and representation of geometric, topological and non-geometric data in this format. The methodology of representing the product definition data in this specification is extensible and independent of the modeling technique being used [1].

IGES is an ANSI standard and has been developed by IGES/PDES organization (IPO) under the auspices of the USPRO, US Product Data Association.

The following IGES entities have been used to transfer the data about the solids with planar boundaries.

Table 5.1 Entity numbers and names of entities used

Entity Type Number	Entity Name
108	Plane Entity
102	Composite Curve Entity
110	Line Entity

The detailed formatting and implementation aspects of IGES have been discussed in Appendix B and a sample file has been shown in Appendix C.

Most of the solid modeling softwares like I-DEAS Master Series® 3.0 and Pro/ENGINEER® Version. 17 can stitch the surfaces together to generate the solid. However, in some CAD systems the solid has to explicitly generated by using a closed quilt. Sometimes the numerical errors in the program might lead to small voids in the surface. These can also be patched up in these softwares. The solid models thus generated can be used for storage, enhancement, analysis or documentation.

Figures 5.4-5.9 show the solid models of the parts whose wire frame models have been shown in Figures 4.4-4.9. Solid models have been automatically generated and transferred to the CAD system. To demonstrate its compatibility with any software, some of the parts have been transferred to I-DEAS Master Series® 3.0 while others are exported to Pro/ENGINEER®.

Solid model of a block is used to generate a range image, as shown in Figure 4.4, using the procedure in Appendix A. Comparison of the original block and the block generated by reverse engineering approach are compared. Table 5.2 compares dimensional, volumetric and mass properties of two blocks.

Table 5.2 Comparison of results of reverse engineering approach for the cube in Figure 5.4

Property	Units	Original Model	Model Generated by Reverse Engineering	% Error
Length (X)	inch	0.5	0.49953	0.094
Breadth (Y)	inch	0.5	0.50039	0.078
Height (Z)	inch	0.5	0.49924	0.152
Volume	$inch^3$	0.125	0.124827	0.1384
Angle between two XY planes	degree	0.0	0.1070	-
Angle between two YZ planes	degree	0.0	0.1024	-
Angle between two XZ planes	degree	0.0	0.1057	-
Density	$lb/inch^3$	1.0 (specified)	1.0 (specified)	-
M.I. about C.G. (I_{xx})	$lb . inch^2$	0.00520833	0.0052026211	0.1094
M.I. about C.G. (I_{yy})	$lb . inch^2$	0.00520833	0.0051883047	0.3786
M.I. about C.G. (I_{zz})	$lb . inch^2$	0.00520833	0.0051981426	0.1957

Some errors were introduced due to numerical errors in computation of eigenvalues, eigenvectors and equations of edges and vertices. Also, since the bounding planes of a part are constructed from the range image data points, the approximation may introduce some errors. However, the errors are very small and hence the procedure is satisfactory. These errors could be eliminated by approximating angles or dimensions by a heuristic method which is used to approximate parallel and perpendicular entities in CAD software. eg. an angle of 0.1 degree would be approximated to 0. This would make the model more realistic.

To estimate the error introduced by the measuring instrument error analysis needs to be done by scanning accurate part such as gauge block and implementing the reverse engineering procedure on the range image generated.

A systematic procedure needs to be developed for study of error propagation through the algorithm so that accuracy of the model generated by reverse engineering could be predicted from the precision of the instrument. This would increase the reliability of the algorithm.

Thus a complete procedure for generating solid CAD models of planar objects from noisy range image has been developed. This is only a first step in development of such an interface. Objects modeled by this algorithm should have all planar faces and no holes. However, feasibility of such a system has been demonstrated.

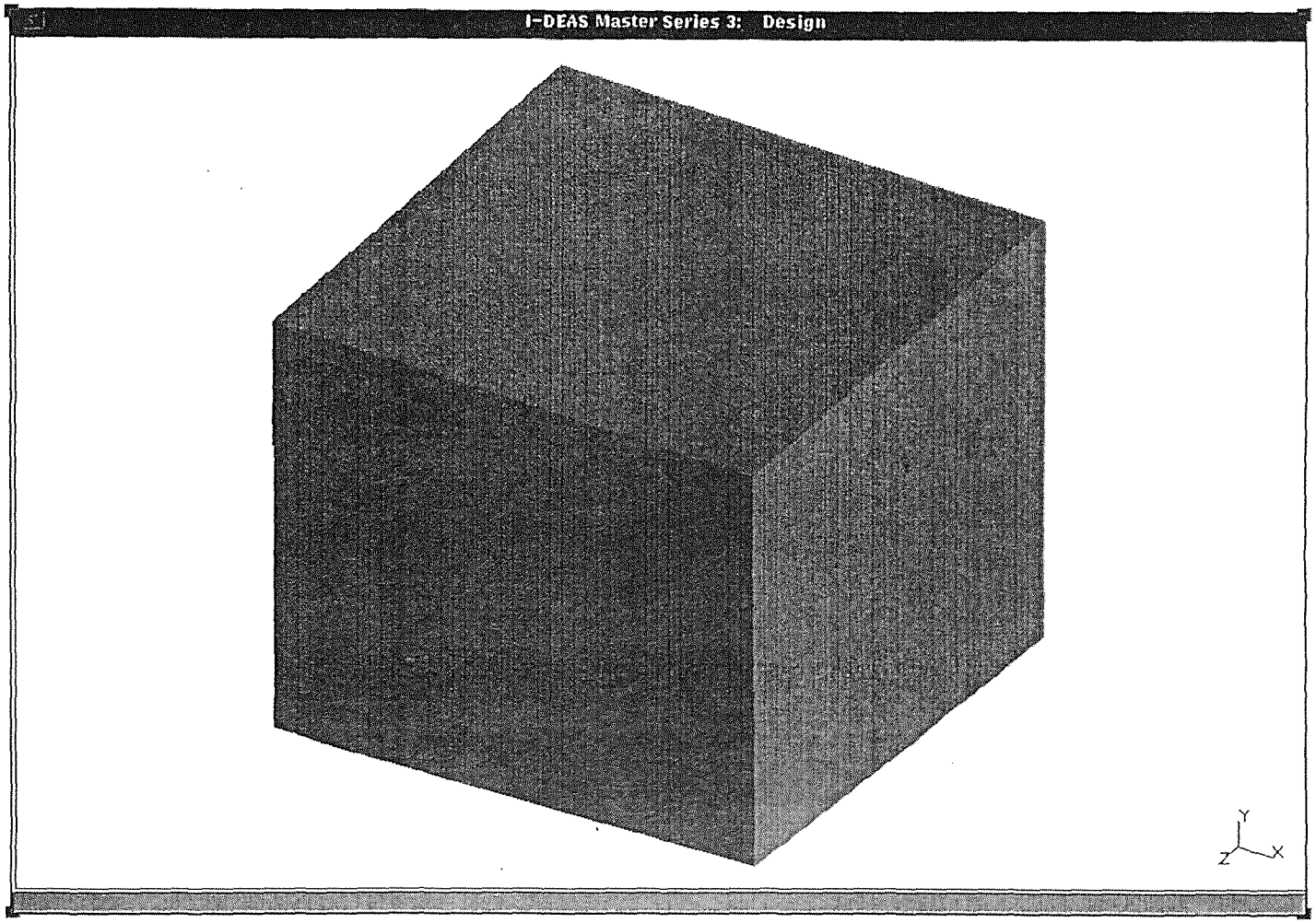


Figure 5.5 Shaded solid model of cube exported to I-DEAS

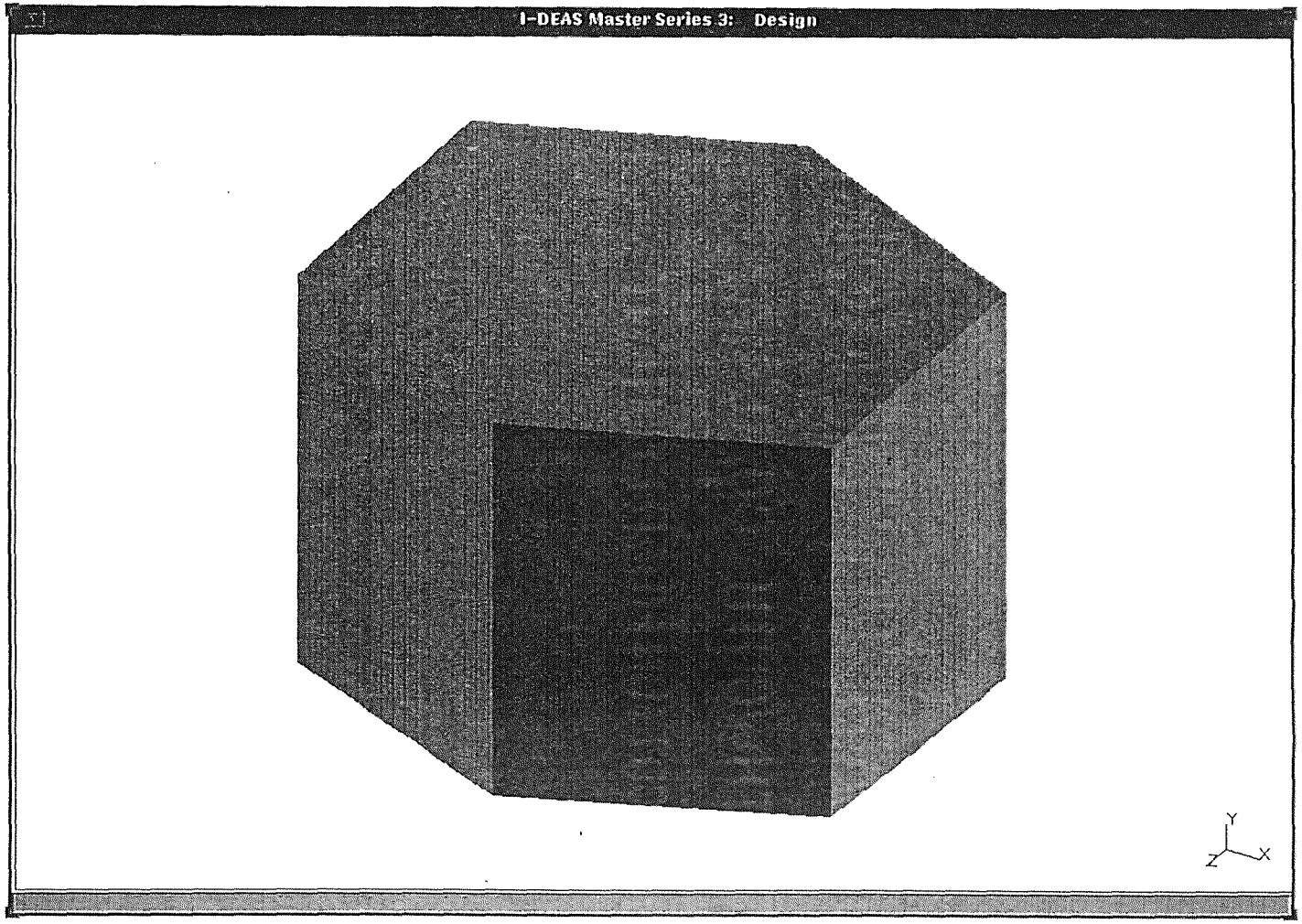


Figure 5.6 Shaded solid model of hexagonal prism exported to I–DEAS

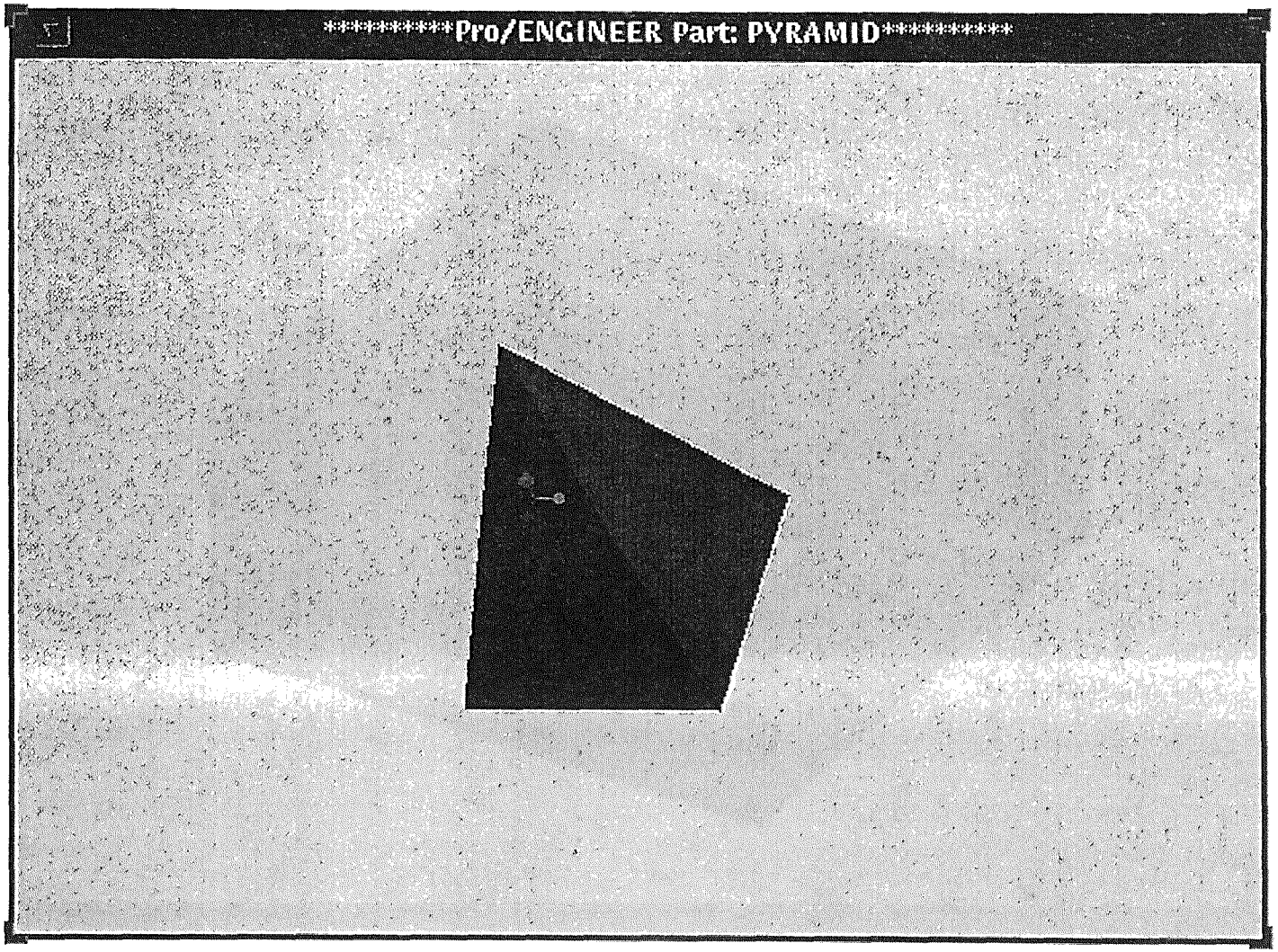


Figure 5.7 Shaded solid model of pyramid exported to Pro/ENGINEER

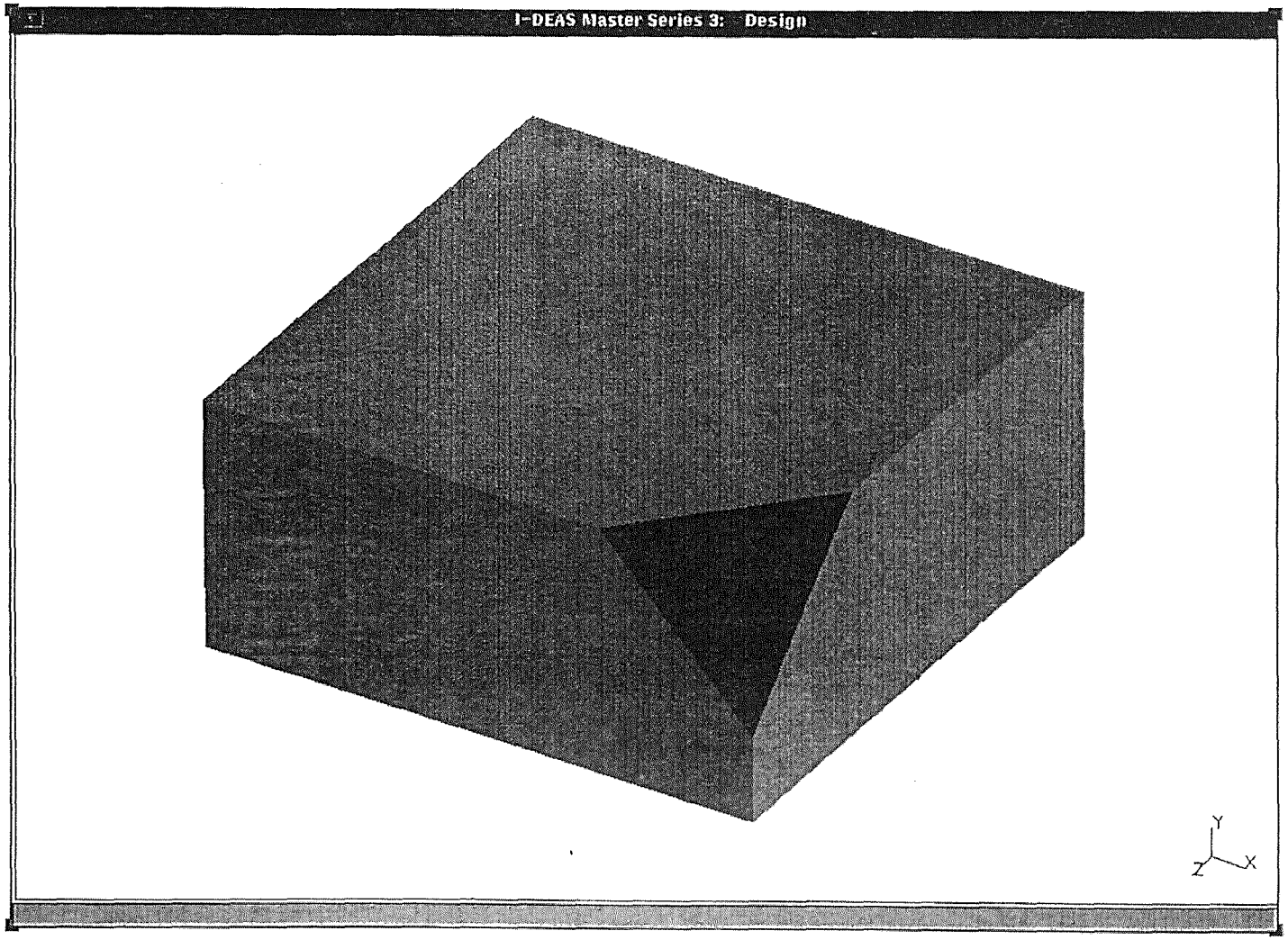


Figure 5.8 Shaded solid model of a chamfered block exported to I-DEAS

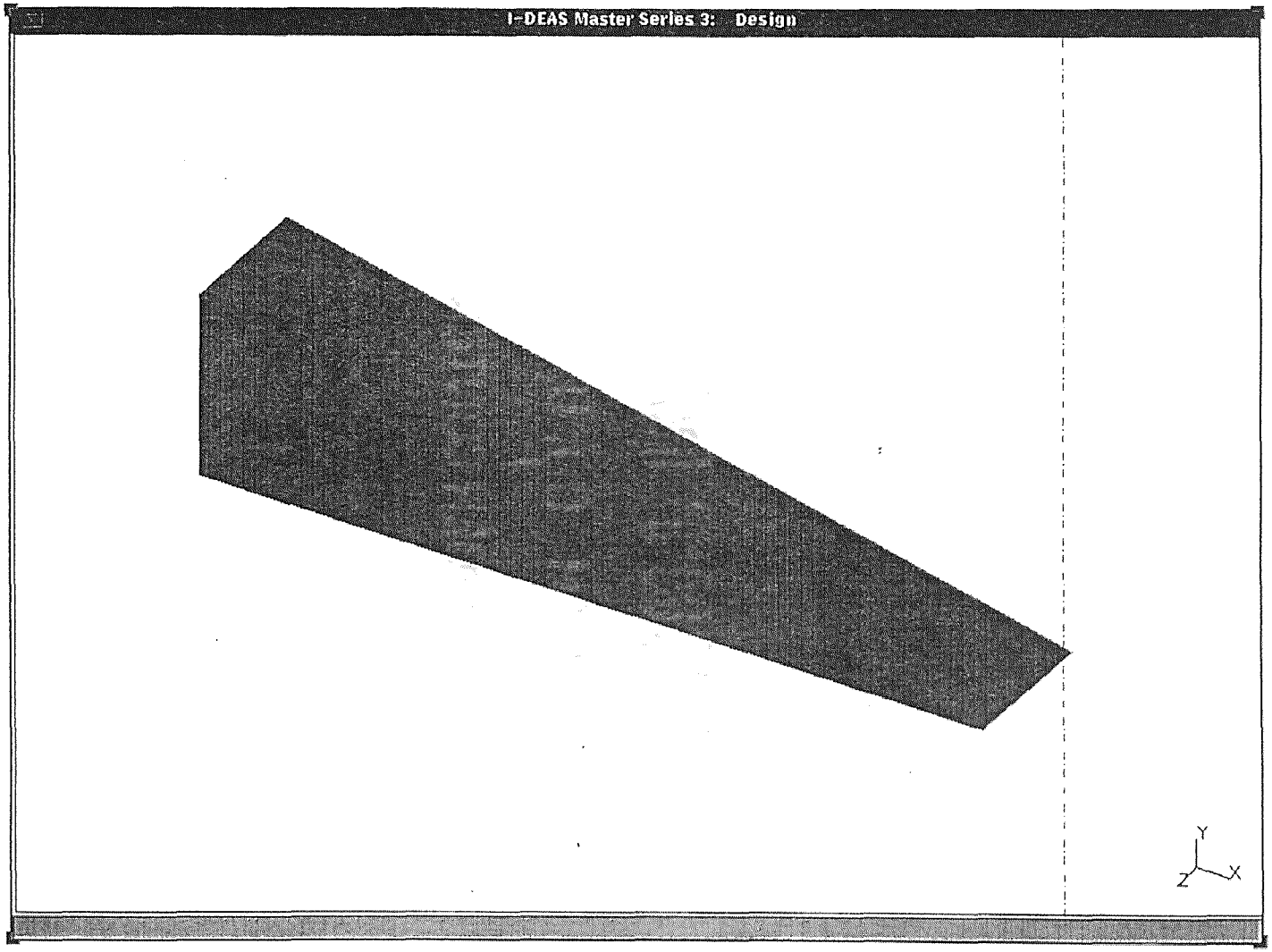


Figure 5.9 Shaded solid model of wedge exported to I-DEAS

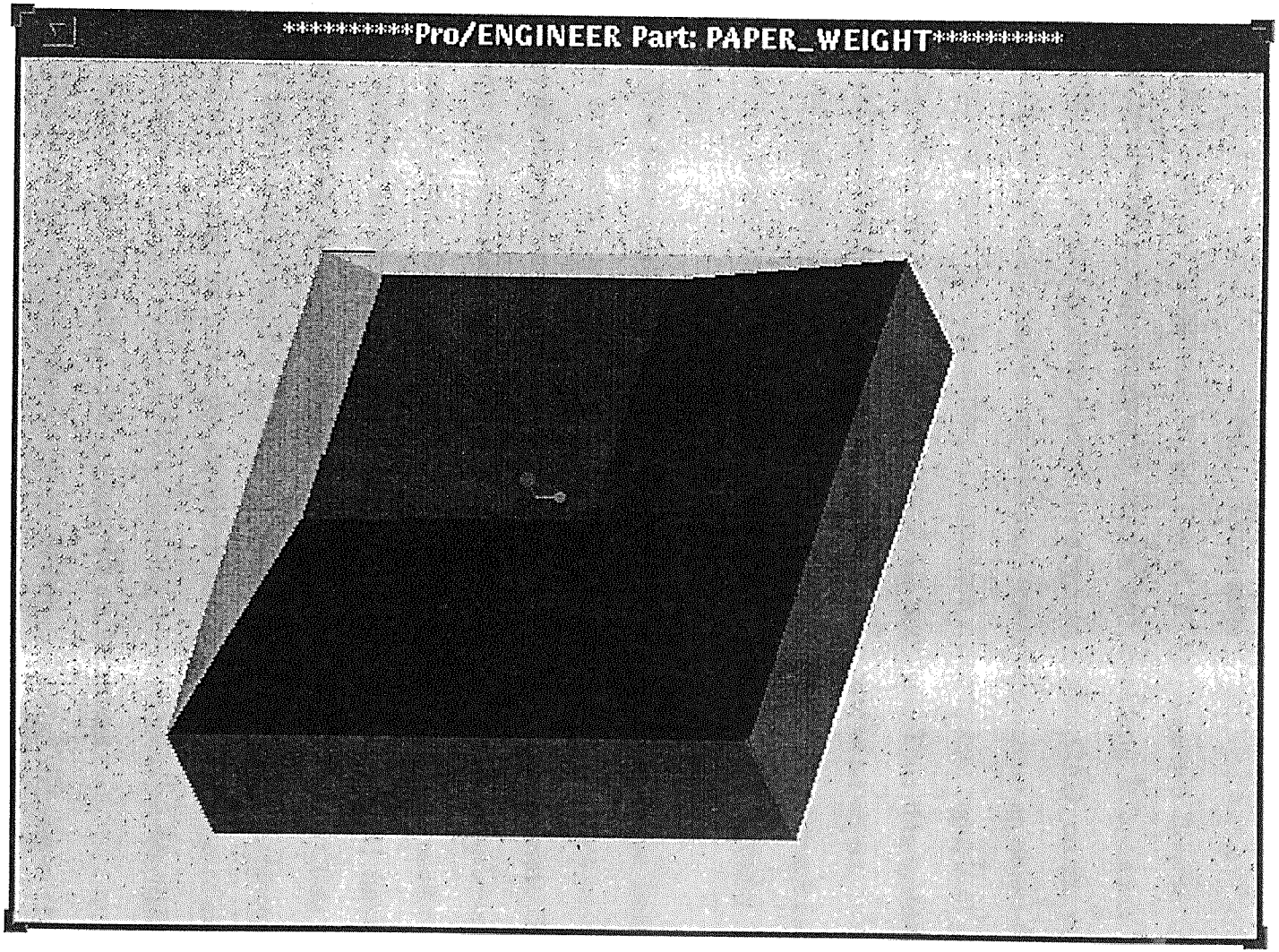


Figure 5.10 Shaded solid model of paper weight exported to Pro/ENGINEER

CHAPTER 6
CONCLUSION

6.1 Discussion

Machine Vision and Computer Aided Design (CAD) are two diverse fields. The former involves scanning of physical objects and extracting information about the geometric and positional properties of the physical object. CAD, on the other hand, deals with the generating and storing electronic information about an object for various purposes. The system developed here is a machine vision system capable of generating the information that can be processed by CAD software. Thus this system forms a bridge between the physical model and the electronic model of an object.

The range image of a physical object is obtained by a range image camera or a co-ordinate measuring machine. It is then processed using the fuzzy clustering algorithms.

The fuzzy clustering algorithms are used to divide the range image into a number of clusters based on some common characteristics of the data points. These involve minimization of an objective functional with some constraints. The fuzzy clustering algorithms can be modified as suggested by Dave [11] to incorporate noise. The Noise Adaptive Fuzzy Clustering Algorithm (NAFC) has been used to find the planar features in the range image. Frigui and Krishnapuram [14] have recently suggested a novel idea of agglomerated competitive fuzzy clustering. This algorithm was not robust with noise. The concept of noise cluster and noise distance has been extended in this work to make the algorithm robust in the presence of noise. The segmented range images obtained from clustering are then processed to obtain object data from the image that can be used to construct a solid model.

Planes are fit on the clusters in the range image. Each of the cluster represents a unique finite planar entity. To define a finite planar entity completely the plane equation and the plane bounding loop has been defined. The plane equation is obtained by the eigenvector approach which is a very effective and computationally

inexpensive way of fitting the planes. It shows better results as compared to the least mean squared approach. The bounding loop is a closed loop of ordered straight edges. The straight line equations are derived from the intersection of two planes. However, not all planes in the body intersect to form an edge. The edge validity criterion is proposed to decide which edges exist in the body. The endpoints of these edges are two points with extreme positions lying on the edge. The guess about the endpoints is further improved by latching them on to the nearest vertices. The vertex positions are obtained by the intersection of three planes. As shown in Table 5.2, this approach has shown satisfactory results for objects without holes.

The information about the solid model is then stored in a data structure for easy accessibility. The data is then transferred to the commercial CAD software using IGES. The model thus generated is ready for processing in CAD softwares.

6.2 Scope of Future Research

The system developed is still in primary stages and has a lot of scope of future research. A generic surface like NURBS could be used to replace the planar surfaces. The free form surfaces can model any object without ambiguity. To achieve this, clustering algorithms for free form surfaces need to be developed. This is very difficult. However, the clustering algorithms can handle ellipsoidal entities. These could be used for modeling the object to improve the range of objects that the system can handle.

Some of the noise points lie on the infinite planes identified by the clustering algorithm and hence interfere during edge delimitation. Some technique to reject such noise points needs to be developed. 2-D clustering techniques could be used after the 3-D clustering to determine the contiguous set of points on a plane. This would reject outliers thus improving the result.

The numerical errors involved may sometimes lead to open surface quilts. Due to this a solid model cannot be generated. A method to reduce the numerical error needs to be developed. The algorithm could be improved by a heuristic approximation

and rounding off such that the faces that are nearly parallel or perpendicular, could be made perfectly parallel or perpendicular.

Propagation of error through the algorithm needs to be studied to make the algorithm more reliable.

APPENDIX A
GENERATION OF PSEUDO RANGE IMAGES

The equipment for the scanning of range images is expensive. The contact type procedures discussed earlier are slow. For testing the algorithm we need a number of range images. A method has been devised to generate range images with added noise.

Most of the CAD softwares are now equipped with the meshing application to make the model useful for FEM analysis. A part is drafted in a CAD software like Pro/ENGINEER. The global maximum for the mesh is set such that the elements are small enough to represent the object correctly. But, it should not be too small, because as the mesh size reduces the number of data points goes up, which in turn means increase in processing time. Once a mesh is generated, it is exported to a neutral file. This neutral file contains the co-ordinates of the nodes along with other information, their connectivity and some other information about the material properties of the object. However, the only information of interest is the position of the nodes.

To extract this information, a program has been written. This program extracts the co-ordinates of the nodes, perturbs each of the points by some distance and adds noise.

The program takes the following input:

1. Input file name. The input file is FEM neutral file generated by the CAD software.
2. Output file name: The output file name is same as the file name for the range image
3. Amount of maximum perturbation for a point.
4. Number of noise points.
5. The radius of sphere in which the noise is randomly distributed.

The data points are perturbed randomly. But their maximum distance from the original position is within the limit defined by the user. The noise points are thrown randomly in a spherical volume of radius specified by the user. The center of this

sphere is the centroid of the data points in their original position. The file thus generated is a pseudo range image and can be used for testing.

APPENDIX B

INITIAL GRAPHICS EXCHANGE SPECIFICATION

Initial Graphics Exchange Specification is available in ASCII as well as Binary form. In this section ASCII format with fixed line length of 80 characters has been discussed.

File Structure:

The file consists of several lines each of length 80 characters. The term "column" refers to the position of a character in a particular line. Throughout the file, the column 73 is occupied by section identifier character. Columns 74 to 80 are reserved for the section sequence number for each line. Six types of sections and their identifiers are given below:

Table B.1 Letter codes for sections

Section	Letter Code
Flag (Not always present)	B or C
Start	S
Global	G
Directory Entry	D
Parameter Data	P
Terminate	T

These sections are contiguous with no intervening blank lines.

Start Section:

The start section of the file is designed to provide human readable prologue to the file. There should be at least one line in the start section. The format of the start section is shown in Fig. B.2

1 72	73 80
This is a human readable prologue	S0000001
Uses ASCII characters in column 1-72	S0000002

Figure B.1 Format of the Start Section

Global Section:

The global section contains information describing the processor and the information needed by the post processor to handle the file. There are in all 25 parameters defined in this section. The first parameter is the parameter delimiter character and the second is the record delimiter character. The next 23 parameters are in free format separated by the parameter delimiters. The section ends with a record delimiter.

Directory Entry Section:

The directory entry section has one directory entry for each entity in the file. Each entity consists of 20 fields each of 8 column width spread across 2 consecutive rows. The entry in each field is right justified.

1 8	9 16	17 24	25 32	33 40	41 48	49 56	57 64	65 72	73 80
(1) Entity Type Number #	(2) Parameter Data ⇒	(3) Structure #, ⇒	(4) Line Font Pattern #, ⇒	(5) Level #, ⇒	(6) View 0, ⇒	(7) Transfor mation Matrix 0, ⇒	(8) Label Display Assoc. 0, ⇒	(9) Status Number #	(10) Sequence Number D#
(11) Entity Type Number #	(12) Line Weight Number #	(13) Color Number #, ⇒	(14) Parameter Line Count #	(15) Form Number #	(16) Reserved	(17) Reserved	(18) Entity Label	(19) Entity Subscript Number #	(20) Sequence number D#+1

Nomenclature:

(n) -Field number n

-Integer

⇒ -Pointer to a directory entry sequence number

#, ⇒ -Integer or pointer to a directory entry sequence number (pointer has a negative sign)

0, ⇒ -Zero or pointer to a directory entry sequence number

Figure B.2 Format of the Directory Entry section

Parameter Data (PD) Section:

Parameter data is placed in a free format with the first field always containing the entity type number. The parameters differ from entity to entity. Each of the entity listed in the directory section should have at least one line in the parameter data section. The parameters are separated by the parameter delimiter declared in the global section. The entry for each entity ends with a record delimiter. Columns 66-72 contain a pointer to the directory entry of the entity.

1 64	66 72	73 80
Entity type number followed by the parameter delimiter followed by parameter separated by parameter delimiters	DE Pointer	P0000001
Parameters separated by parameter delimiters followed by section delimiters	DE Pointer	P0000002

Figure B.3 Format for the Parameter Data (PD) Section

Terminate Section:

Terminate section has only one line. It contains total number of lines in each of the previous sections. The line is divided into 10 sections of 8 characters each. The first character in each of the first four fields is the section-identification character and the next 7 characters represent the last sequence number used in each of the sections.

1 8	9 16	17 24	25 32	33 72	73 80
S0000005	G0000003	D0000100	P0000073	Not used	T0000001

Figure B.4 Format of the Terminate Section.

Entities used and their Parameters:

1. Plane Entity (Type 108): The plane is defined by $Ax+By+Cz=D$. The parameters are:
   - Coefficient A
   - Coefficient B
   - Coefficient C
   - Coefficient D
   - Pointer to a closed curve entity for bounding the plane or 0 for unbounded.
   - x co-ordinate for display symbol.
   - y co-ordinate for display symbol.
   - z co-ordinate for display symbol.
2. Composite Curve Entity (Type 102): This curve is formed by connecting different curve entities.
   - Number of entities
   - Pointer to the DE of the constituent entities separated by delimiters.
3. Line Entity (Type 110): The line connecting points (X1,Y1,Z1) and (X2,Y2,Z2). Parameters are:
   - X1
   - Y1
   - Z1
   - X2
   - Y2
   - Z2.

APPENDIX C

IGES FILE SAMPLE

The following sample IGES file. This file has been written to transfer the solid model of the cube in Figure 5.4.

```
This file is a test file written by the program to check whether ProE   S0000001
1H,,1H;,6HManual,7Hone.igs,10HPhansalkar,8HIGES 5.2,32,38,7,38,15,      G0000001
10HPhansalkar,1.,1,4HINCH,32768,0.5,13H961022.154500,0.0001000,         G0000002
1.732,4HNONE,4HNONE,9,0;                                                G0000003
     110       1       1       1       0       0       0       0000100000D       1
     110       0       0       1       0                             LINE       1D       2
     110       2       1       1       0       0       0       0000100000D       3
     110       0       0       1       0                             LINE       2D       4
     110       3       1       1       0       0       0       0000100000D       5
     110       0       0       1       0                             LINE       3D       6
     110       4       1       1       0       0       0       0000100000D       7
     110       0       0       1       0                             LINE       4D       8
     102       5       1       1       0       0       0       0000100000D       9
     102       0       0       1       0                           CCURVE       1D      10
     108       6       1       1       0       0       0       0000000000D      11
     108       0       0       1       1                           PLANE        1D      12
     110       7       1       1       0       0       0       0000100000D      13
     110       0       0       1       0                             LINE       7D      14
     110       8       1       1       0       0       0       0000100000D      15
     110       0       0       1       0                             LINE       8D      16
     110       9       1       1       0       0       0       0000100000D      17
     110       0       0       1       0                             LINE       9D      18
     110      10       1       1       0       0       0       0000100000D      19
     110       0       0       1       0                             LINE      10D      20
     102      11       1       1       0       0       0       0000100000D      21
     102       0       0       1       0                           CCURVE       2D      22
     108      12       1       1       0       0       0       0000000000D      23
     108       0       0       1       1                           PLANE        2D      24
     110      13       1       1       0       0       0       0000100000D      25
     110       0       0       1       0                             LINE      13D      26
     110      14       1       1       0       0       0       0000100000D      27
     110       0       0       1       0                             LINE      14D      28
     110      15       1       1       0       0       0       0000100000D      29
     110       0       0       1       0                             LINE      15D      30
     110      16       1       1       0       0       0       0000100000D      31
     110       0       0       1       0                             LINE      16D      32
     102      17       1       1       0       0       0       0000100000D      33
     102       0       0       1       0                           CCURVE       3D      34
     108      18       1       1       0       0       0       0000000000D      35
     108       0       0       1       1                           PLANE        3D      36
     110      19       1       1       0       0       0       0000100000D      37
     110       0       0       1       0                             LINE      19D      38
     110      20       1       1       0       0       0       0000100000D      39
     110       0       0       1       0                             LINE      20D      40
     110      21       1       1       0       0       0       0000100000D      41
     110       0       0       1       0                             LINE      21D      42
     110      22       1       1       0       0       0       0000100000D      43
     110       0       0       1       0                             LINE      22D      44
     102      23       1       1       0       0       0       0000100000D      45
     102       0       0       1       0                           CCURVE       4D      46
     108      24       1       1       0       0       0       0000000000D      47
     108       0       0       1       1                           PLANE        4D      48
     110      25       1       1       0       0       0       0000100000D      49
     110       0       0       1       0                             LINE      25D      50
     110      26       1       1       0       0       0       0000100000D      51
     110       0       0       1       0                             LINE      26D      52
     110      27       1       1       0       0       0       0000100000D      53
     110       0       0       1       0                             LINE      27D      54
     110      28       1       1       0       0       0       0000100000D      55
```

```
     110       0       0       1       0                       LINE       28D      56
     102      29       1       1       0       0       0          0000100000D      57
     102       0       0       1       0                     CCURVE        5D      58
     108      30       1       1       0       0       0          0000000000D      59
     108       0       0       1       1                      PLANE        5D      60
     110      31       1       1       0       0       0          0000100000D      61
     110       0       0       1       0                       LINE       31D      62
     110      32       1       1       0       0       0          0000100000D      63
     110       0       0       1       0                       LINE       32D      64
     110      33       1       1       0       0       0          0000100000D      65
     110       0       0       1       0                       LINE       33D      66
     110      34       1       1       0       0       0          0000100000D      67
     110       0       0       1       0                       LINE       34D      68
     102      35       1       1       0       0       0          0000100000D      69
     102       0       0       1       0                     CCURVE        6D      70
     108      36       1       1       0       0       0          0000000000D      71
     108       0       0       1       1                      PLANE        6D      72
110,0.499286,0.500797,0.001409,0.499388,-0.000391,0.000000;                        1P       1
110,0.499388,-0.000391,0.000000,-0.000140,-0.000925,-0.000857;                     3P       2
110,-0.000140,-0.000925,-0.000857,-0.000397,0.499465,0.000549;                     5P       3
110,-0.000397,0.499465,0.000549,0.499286,0.500797,0.001409;                        7P       4
102,4,1,3,5,7;                                                                     9P       5
108,0.001713,0.002811,-0.999995,0.000854,9,0.0,0.0,0.0,1.0;                       11P       6
110,0.498878,0.501187,0.501366,0.498980,0.000490,0.499385;                        13P       7
110,0.498980,0.000490,0.499385,0.000326,-0.000044,0.498380;                       15P       8
110,0.000326,-0.000044,0.498380,0.000070,0.499857,0.500357;                       17P       9
110,0.000070,0.499857,0.500357,0.498878,0.501187,0.501366;                        19P      10
102,4,13,15,17,19;                                                                21P      11
108,-0.002012,-0.003956,0.999990,0.498374,21,0.0,0.0,0.0,1.0;                     23P      12
110,0.499286,0.500797,0.001409,0.499388,-0.000391,0.000000;                       25P      13
110,0.499388,-0.000391,0.000000,0.498980,0.000490,0.499385;                       27P      14
110,0.498980,0.000490,0.499385,0.498878,0.501187,0.501366;                        29P      15
110,0.498878,0.501187,0.501366,0.499286,0.500797,0.001409;                        31P      16
102,4,25,27,29,31;                                                                33P      17
108,-1.000000,-0.000201,-0.000818,-0.499388,33,0.0,0.0,0.0,1.0;                   35P      18
110,-0.000397,0.499465,0.000549,-0.000140,-0.000925,-0.000857;                    37P      19
110,-0.000140,-0.000925,-0.000857,0.000326,-0.000044,0.498380;                    39P      20
110,0.000326,-0.000044,0.498380,0.000070,0.499857,0.500357;                       41P      21
110,0.000070,0.499857,0.500357,-0.000397,0.499465,0.000549;                       43P      22
102,4,37,39,41,43;                                                                45P      23
108,-0.999999,-0.000517,0.000935,0.000140,45,0.0,0.0,0.0,1.0;                     47P      24
110,-0.000140,-0.000925,-0.000857,0.499388,-0.000391,0.000000;                    49P      25
110,0.499388,-0.000391,0.000000,0.498980,0.000490,0.499385;                       51P      26
110,0.498980,0.000490,0.499385,0.000326,-0.000044,0.498380;                       53P      27
110,0.000326,-0.000044,0.498380,-0.000140,-0.000925,-0.000857;                    55P      28
102,4,49,51,53,55;                                                                57P      29
108,0.001066,-0.999998,0.001765,0.000923,57,0.0,0.0,0.0,1.0;                      59P      30
110,-0.000397,0.499465,0.000549,0.499286,0.500797,0.001409;                       61P      31
110,0.499286,0.500797,0.001409,0.498878,0.501187,0.501366;                        63P      32
110,0.498878,0.501187,0.501366,0.000070,0.499857,0.500357;                        65P      33
110,0.000070,0.499857,0.500357,-0.000397,0.499465,0.000549;                       67P      34
102,4,61,63,65,67;                                                                69P      35
108,0.002664,-0.999996,0.000783,-0.499464,69,0.0,0.0,0.0,1.0;                     71P      36
S      1G      3D     72P     36                                                          T0000001
```

Figure C.1 IGES sample file

REFERENCES

1. *Digital Representation for Communication of Product Definition Data: USPRO/IPO 100 IGES 5.2*, U.S. Product Data Association, Fairfax, VA, 1993.

2. Abella R.J., Daschbach J.M. and McNichols R.J., "Reverse Engineering Industrial Engineering", *Computers and Industrial Engineering*, Vol. 26, No. 2, pp. 381 385, 1994.

3. Anand S. and Knott K., "An Algorithm for converting the Boundary Representation of a CAD Model to its Octree Representation", *Computers and Industrial Engineering*, Vol. 21, No. 1-4, pp. 343-347, 1991.

4. Aronson R.B, "Forward thinkers take to reverse engineering", *Manufacturing Engineering*, pp. 34-44, November 1996.

5. Atre A., *Detection of Planar Facets in Noisy Range Images*, M.S. Thesis, Department of Mechanical Engineering, New Jersey Institute of Technology, Newark, NJ, 1993.

6. Bezdek J.C., *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, New York, 1981.

7. Broacha R. and Young M., "Getting from Points to Products", *Computer-Aided Engineering*, Vol. 14(7), pp. 28-32, 1995.

8. Burden R.L. and Faires J.D., *Numerical Analysis*, 5th Ed., PWS Publishing Company, Boston, MA, 1993.

9. Dave R.N., "An Adaptive Fuzzy c-Elliptotype clustering algorithm", *Proc. of the NAFIPS 90: Quarter Century of Fuzziness*, ed. I.B. Turksen, Vol I, pp.9-12, 1990.

10. Dave R.N., "Boundary Detection through Fuzzy Clustering", Invited paper, *IEEE International conference on Fuzzy Systems*, San Diego, California, pp.127-134, March 8-12, 1992.

11. Dave R.N., "Characterization and Detection of Noise in Clustering", *Pattern Recognition Letters*, Vol 12(11), pp. 657-664, 1991.

12. Dave R.N., "Use of the adaptive fuzzy clustering algorithm to detect lines in digital images", *Intelligent Robots and Computer Vision VIII*, Vol. 1192(2), pp. 600-611, 1989.

13. Dave R.N. and Patel K.J., "Progressive Fuzzy Clustering Algorithm for characteristic shape recognition", *Proc. of the NAFIPS 90: Quarter Century of Fuzziness*, ed. I.B. Turksen, Vol I, pp.121-124, 1990.

14. Frigui H. and Krishnapuram R., "A robust clustering algorithm based on competitive agglomeration and soft rejection of outliers", *Proc. of IEEE conference on Computer Vision and Pattern Recognition*, San Fransisco, California, 1996.

15. Goldman R.N., "The role of surfaces in solid modeling", *Geometric Modeling Algorithms and New Trends*, pp. 69-90, 1987.

16. Gunderson R., "An Adaptive FCV clustering algorithm", *Int. Journal of Man Machine Studies*, pp.97-104, 1983.

17. Hosser Y. and Ferreira L., "Laser based system for reverse engineering". *Computers and Industrial Engineering*, Vol. 26, No. 2, pp. 387-394, 1994.

18. Huang C. and Motavalli S., "Reverse Engineering of Planar Parts using Machine Vision", *Computers and Industrial Engineering*, Vol. 26, No. 2, pp. 369-379, 1994.

19. Kreyszig E., *Advanced Engineering Mathematics*, 5th Ed., Wiley Eastern Limited, New Delhi, 1994.

20. Krishnapuram R. and Freg C.P., "Fitting an unknown number of lines and planes to image data through compatible cluster merging", *Pattern Recognition*, 1991.

21. Krishnapuram R., Frigui H. and Nasraoui O., "Fuzzy and Possibilistic Shell Clustering Algorithms and their application to boundary detection and surface approximation-Part I", *IEEE Transactions on Fuzzy systems*, Vol. 3, No. 1, pp. 29-43, February 1995.

22. Kwok W. and Eagle P.J., "Reverse Engineering: Extracting CAD Data From Existing Parts", *Mechanical Engineering*, pp. 52-55, March 1991.

23. Mortenson M.E., *Geometric Modeling*, John Wiley & Sons, Inc., New York, 1985.

24. Motavalli S. and Bidanda B., "Modular software development for digitizing systems data analysis in reverse engineering applications: Case of concentric rotational parts", *Computers and Industrial Engineering*, Vol. 26, No. 2, pp. 395-410, 1994.

25. Phansalkar G. and Dave R.N., "On Generating Solid Models of Mechanical Parts through Fuzzy Clustering", submitted, *Sixth IEEE Conference on Fuzzy systems, (FUZZ-IEEE'97),* Barcelona, Spain, July 1-5, 1997.

26. Rogers D.F. and Adams J.A., *Mathematical Elements of Computer Graphics*, 2nd Ed., McGraw Hill Publishing Co., New York, 1990.

27. Zeid I., *CAD / CAM theory and practice*, McGraw-Hill. Inc, New York, 1993.